AIN SHAMS UNIVERSITY
FACULTY OF ENGINEERING
Computer Engineering and Systems

Metagenomic data analysis using deep learning

A Thesis submitted in partial fulfillment of the requirements of
Master of Science in Electrical Engineering
(Computer Engineering and Systems)

by

Aly O. Abdelkareem

Bachelor of Science in Electrical Engineering
(Computer Engineering and Systems)
Faculty of Engineering, Ain Shams University, 2016

Supervised By

Prof. Hazem M. Abbas

Dr. Mahmoud I. Khalil

Cairo, 2018

AIN SHAMS UNIVERSITY
FACULTY OF ENGINEERING
Computer Engineering and Systems

Metagenomic data analysis using deep learning

by

Aly O. Abdelkareem

Bachelor of Science in Electrical Engineering
(Computer Engineering and Systems)
Faculty of Engineering, Ain Shams University, 2016

Examiners' Committee

Name and affiliation	**Signature**
Prof. Computer Engineering and Systems Faculty of Engineering, Ain Shams University.	.....
Prof. Computer Engineering and Systems Faculty of Engineering, Ain Shams University.	.....
Dr. Choose Department Faculty of Engineering, University.	.....

Date:25 Feb 2019

Statement

This thesis is submitted as a partial fulfillment of Master of Science in Electrical Engineering, Faculty of Engineering, Ain Shams University. The author carried out the work included in this thesis, and no part of it has been submitted for a degree or a qualification at any other scientific entity.

Aly O. Abdelkareem

Signature

.....

Date: 25 Dec 2018

Researcher Data

Name: Aly Osama Aly Ibrahim Abdelkareem (Aly O. Abdelkareem)
Date of Birth: 29/08/1992
Place of Birth: Cairo, Egypt
Last academic degree: Bachelor of Science in Electrical Engineering
Field of specialization: Computer Engineering and Software Systems (Credit Hours)
University issued the degree : Ain Shams University
Date of issued degree : 15/6/2016
Current job : Teaching and Research Assistant at Faculty of Engineering

Thesis Summary

Summary

Metagenomics shows a promising understanding of function and diversity of the microbial communities due to the difficulty of studying microorganism with pure culture isolation. Moreover, the viral identification is considered one of the essential steps in studying microbial communities. Several studies show different methods to identify viruses in mixed metagenomic data and phages in host genomes, using homology and statistical techniques. These techniques have many limitations due to viral genome diversity. In this work, we propose a sequence deep neural model for viral identification of metagenomic data. For testing purpose, we generated fragments of viruses and bacteria from RefSeq genomes with different lengths to find the best hyperparameters of our model. Then, we simulated both microbiome and virome high throughput data from our test genomes dataset with aim of validating our approach. Finally, we applied our tool to a case study of two types of metagenomic data such as Roche 454 and Illumina. We compared our tool to the state-of-the-art statistical and popular tool for viral identification and found the performance of VirNet much better regarding accuracy and speed on the same testing data. This tool will help us in growing our insights to natural viruses of microbial communities.

Key words: bioinformatics, classification, deep learning, metagenomics

Acknowledgment

Aly O. Abdelkareem
Computer Engineering and Systems
Faculty of Engineering
Ain Shams University
Cairo, Egypt
Dec 2018

I would like to express my sincere gratitude to my mother and my father for the continuous support of my study, my wife Mayada for her patience and motivation, and my daughter Aisha as she brought happiness to my life.

Contents

List of Figures

List of Tables

Abbreviations

LAH **L**ist **A**bbreviations **H**ere

Symbols

a	distance	m
P	power	W (Js^{-1})
ω	angular frequency	rads^{-1}

Chapter 1

Introduction

1.1 Metagenomic analysis

1.1.1 Definition

Metagenomics is an analysis of the genetic information of the collective genomes of the microbes within a given environment based on its sampling regardless of cultivability of the cells. [1]. There is a minor population of microbial organisms identified due to the difficulty in studying them using pure culture isolation. This methodology has been constrained to less than 1% of host cells and is biased to certain species [2]. Metagenomic analysis process demonstrates a promising understanding of different microorganisms. It answers some questions about the identity of microorganisms in the collection and their potential functional characterization.

1.1.2 Microorganism

Microorganisms are found everywhere on earth, and they are critical in our survival. This study, our interest is in prokaryotic microorganisms (e.g. bacteria and archaea) and viruses. Bacteria are unicellular and microscopic organisms that reproduce by binary fission. On the other hand, viruses are typically submicroscopic consists of genetic materials either DNA or RNA surrounded by a protective coat of proteins and can only replicate inside living host cells. They lack metabolic enzymes and translational machinery such as ribosomes for making proteins. There are 200 to a few thousand

genes in the bacterial genomes, while the tiniest viral genomes have only three genes and the largest have up to 2000 genes.

1.2 Viruses

1.2.1 Definition

Viruses have an impact on different microbial communities, and virus-host interaction can change many ecosystems such as human health and aquatic life. Phages or bacteriophages are viruses that infect bacteria. Furthermore, phages are abundant in different microbiome communities. The viral infection starts when virus binds to a host cell and its genome integrates with the host cell genome. The integrated viral DNA is called a provirus. It is reasonable to think that isolated viruses are just package of genes moving from one host cell to another.

1.2.2 Importance in clinical and environment

very very important.

1.2.3 Identification

Scientists are using isolation and culturing techniques to study viral diversity and viral-host interactions in microbial communities. Those techniques have many limitations because there is no universal marker gene for viruses. The sequenced viruses in NCBI RefSeq database constitute approximately 5% of known species of prokaryotic organisms [3].

1.3 Next Generation Sequencing

High throughput sequencing technology is used for metagenomic studies which can generate large number of read sequences of microorganisms. The expected read length is up to 600 bp and the number of generated reads per run is up to 15 million approximately

based on the sequencing platform and the library preparation methods [4]. We can sequence mixture of prokaryotic cells and viruses in complex microbial communities in a cultivation-independent process. Sequencing of microbial samples shows contamination of viral sequences within prokaryotic population. A study found 4-17% virus sequences in human gut prokaryotic metagenomes [5]. Moreover, cellular contamination is quite frequent even with a careful purification of viral particles, and this is one of the main reasons why we need a tool that can differentiate between bacterial and viral sequences.

1.3.1 Sequencer Tools

different tools of HTS

1.3.2 Data types

different types of data

1.3.3 Sequence identification in HTS

The broadly adopted technique to know who is in metagenomic data is to assemble the high throughput reads to contigs then search against a known genomic database using sequence alignment method in order to infer the type of microorganisms and the existence of species in a metagenomic sample. This approach is minimal because it only detects viruses almost related to those we already know. It is reported that about 15% of viruses in the human gut microbiome and 10% in the ocean have similarity to the known viruses [6].

1.4 Machine learning

Machine learning approaches have been used to classify and cluster data based on extracted features. The deep neural network is one of machine learning methods that are considered as a state of the art category for general classification problems. Deep learning shows significant improvements in several artificial intelligence tasks for example

image classification, speech recognition, and natural language processing. Moreover, It shows significant results with genomic data [7].

1.5 Our Contribution

In this paper, we introduce a deep sequence model, VirNet, to identify viral reads from a mixture of viral and bacterial sequences and purify viral metagenomic data from bacterial contamination as well. That will guide us to identify new viruses and potentially perform functional characterization. Additionally, it will answer many mysteries related to our understanding of their functionality and diversity in the ecosystem.

Chapter 2

Related Work

There has been extensive prior work on viral identification. Recent work has focused on identifying phages in bacterial genomes. Several methods have used similarity search by sequence alignment with the reference genomes in order to find viral contigs. Most of the recent tools fall under three categories based on the sample structure such as:

1. phages from prokaryotic genomes
2. viral sequences in mixed metagenomic datasets
3. phages and viral sequences.

2.1 Similarity tools

There are many software packages to find phages from prokaryotic genomes such as Phage_Finder [8], Prophinder [9], PHAST[10], and PhiSpy [11]. These tools are using similarity search to known virus databases using features such as genes. Some of them such as PhiSpy integrates other features such as unique virus k-mers, AT and GC skew, protein length and transcription strand direction. They have many limitations as they failed to detect viral sequences in metagenomic data as the databases are outdated, limited and don't represent viral diversity in the environment. Moreover, It is not optimized to process a large number of contigs [12] as they depend on alignment and homology processing limitations.

The second category is able to detect viral sequences in mixed metagenomic datasets such as VIROME [13] and MetaVir[14]. They are using similarity search with the databases same as the first category. Additionally, they are searching against proteins. There are more packages such as DIAMOND [15] or Centrifuge [16] which are much faster and efficient than the former tools for microbial classification. Again, The limitation of this approach is using limited known reference databases.

2.2 Statistical tools

The third category of software packages such as VirSorter [12] is able to detect phages and viral sequences. VirSorter is using similarity search to viral databases and integrates other features related to analysis of sequence genes such as enrichment of viral-like genes, enrichment of uncharacterized genes and viral hallmark gene. These features make the identification more accurate but it still suffer limitations. One of the limitations is the requirements of having at least 3 genes within the contig because the smallest virus genome contains 3 genes because the smallest virus discovered has 3 genes only so it has the same limitations as previous techniques because of using homology strategy. Moreover, it cannot work with short fragments or contigs and it is very slow in processing metagenomic datasets.

Recently VirFinder [6] applied machine learning techniques. VirFinder is a statistical method based on the logistic regression model. It uses the K-mer feature which is considered as a discrimination feature for different sequence problems. It shows a great success with short sequences too and They found a great k-mer similarity score with viruses within other prokaryotic genomes.

In this paper, we are using deep learning techniques which is much more suitable to sequence problems and also shows significant improvements to other current machine learning models. In deep neural networks, the model will extract the most appropriate features during training which lead to better identification accuracy and sensitivity.

Chapter 3

Deep neural networks for identification

3.1 Sequence neural networks

Recurrent neural networks (RNN), long short-term memory (LSTM) [17] and gated recurrent neural networks (GRU) [18] can model complex sequences and have been used for sequence modeling problems.

Our deep learning model is implemented as an attentional encoder network (Figure 3.1a). An input sequence $\mathbf{x} = (\mathbf{x_1}, \ldots, \mathbf{x_m})$ and calculates a forward sequence of hidden states $(\overrightarrow{h_1}, \ldots, \overrightarrow{h_m})$. The hidden states $\overrightarrow{h_j}$ are averaged to obtain the attention vector $\mathbf{h_j}$ representing the context vector from the input sequence.

Embedding layer maps discrete input words to dense vectors for computational efficiency before feeding this sequence to LSTM/GRU Layers. The attentional network could learn how to extract suitable features from the raw data and can attend to previous DNA nucleotide within the same input sequence.

LSTM encoder have 3 gates to protect and control the cell state, the input gate denoted **i** which defines how much of the newly computed state you want to let through, forget gate denoted **f** that decides what information is to be kept and what is to be thrown away, the output of the update gate denoted **U** that's used to update the cell state and the output of the LSTM cell **o** .**W** is the recurrent connection at the previous hidden

layer and current hidden layer and **C** is the internal memory of the unit as shown in the following equations

$\mathbf{i_t} = \mathbf{œ}(\mathbf{x_t U^i} + \mathbf{h_{t-1} W^i})$

$\mathbf{f_t} = \mathbf{œ}(\mathbf{x_t U^f} + \mathbf{h_{t-1} W^f})$

$\mathbf{o_t} = \mathbf{œ}(\mathbf{x_t U^o} + \mathbf{h_{t-1} W^o}).$

GRU encoder is same as LSTM except it has only 2 gates, Reset gate denoted **r** that determines how to combine the new input with the previously saved input state and the update gate denoted **z** that defines the amount of information to keep around, as defined in the following equations

$\mathbf{z_t} = \mathbf{œ}(\mathbf{x_t U^z} + \mathbf{h_{t-1} W^z})$

$\mathbf{r_t} = \mathbf{œ}(\mathbf{x_t U^r} + \mathbf{h_{t-1} W^r})$

$\overline{\mathbf{h_t}} = \mathbf{tanh}(\mathbf{x_t U^h} + (\mathbf{r_t} * \mathbf{h_{t-1}})\mathbf{W^h})$

$\mathbf{h_t} = (\mathbf{1} - \mathbf{z_t})\mathbf{h_{t-1}} + \mathbf{z_t}\overline{\mathbf{h_t}}.$

The attentional neural model was trained with the DNA nucleotide bases with fragments with different lengths. The model will predict in a binary output format whether this fragment is viral or non-viral.

The top-performing model (Figure 3.1b) consists of an input embedding layer of size 128 mapping input DNA nucleotide tokens into an embedding space, that is fed to an LSTM layer. The forward sequence $\overrightarrow{h_j}$ is then averaged together to create an attentional vector representing token context within the same fragment. A dropout layer was added after the attentional layer to avoid overfitting over the input data.

LSTM layer was performing better as in 3.1 than the GRU cell. GRU encoder having less gates than LSTM model make it faster and easier to converge, but depedending on the size and the format of the input data. LSTM with more gates would be slower but will outperform GRU encoder type.

The input sequence is divided into 5 grams sized tokens these tokens are then treated as a single word (Figure 3.1a). This single token is mapped as a point in the embedding space created during training the neural model.

During training, all parameters are optimized jointly using Adam to maximize the conditional probability of tokens found together to predict if an input sequence is viral or not.

In this model, an early stopping mechanism was used as a form of regularization to avoid over-fitting over the data while making more epochs over the data. The early stopping mechanism was used with patience of 3 non -improving consecutive epochs; the neural model will stop training while saving the latest improving checkpoint over the validation set defined.

3.1.1 Attention mechanism

LaTeX is not a WYSIWYG (What You See is What You Get) program, unlike word processors such as Microsoft Word or Apple's Pages. Instead, a document written for LaTeX is actually a simple, plain text file that contains *no formatting*. You tell LaTeX how you want the formatting in the finished document by writing in simple commands amongst the text, for example, if I want to use *italic text for emphasis*, I write the '`\textit{}`' command and put the text I want in italics in between the curly braces. This means that LaTeX is a "mark-up" language, very much like HTML.

3.2 Convolution neural networks

LaTeX is not a WYSIWYG (What You See is What You Get) program, unlike word processors such as Microsoft Word or Apple's Pages. Instead, a document written for LaTeX is actually a simple, plain text file that contains *no formatting*. You tell LaTeX how you want the formatting in the finished document by writing in simple commands amongst the text, for example, if I want to use *italic text for emphasis*, I write the '`\textit{}`' command and put the text I want in italics in between the curly braces. This means that LaTeX is a "mark-up" language, very much like HTML.

3.2.1 Triplet loss mechanism

LaTeX is not a WYSIWYG (What You See is What You Get) program, unlike word processors such as Microsoft Word or Apple's Pages. Instead, a document written for LaTeX is actually a simple, plain text file that contains *no formatting*. You tell LaTeX how you want the formatting in the finished document by writing in simple commands amongst the text, for example, if I want to use *italic text for emphasis*, I write the

‘\textit{}’ command and put the text I want in italics in between the curly braces. This means that LaTeX is a “mark-up” language, very much like HTML.

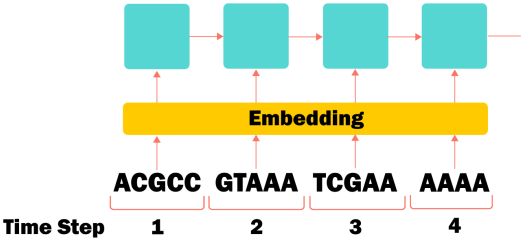
3.2.2 Hyperparameters optimization

Model parameters affect the performance of the deep learning model, and they control the behavior of the training algorithm as well. We selected the grid search technique in order to find the most suitable parameters. The grid search is considered a traditional technique for hyperparameters optimization and it brute force different combinations. We ran several experiments on 20% of our training set for 500 bp. Then, we divided it into training, validation, and testing set with the following percentages 70%, 10%, and 20%. These experiments were designed to find the best parameters for the number of recurrent layers, the embedding size for each layer and the input sub-words (ngram). Our reported results (Table 3.1) show that the best parameters setup is for 2 layers, 128 embedding size, and 5 ngram.

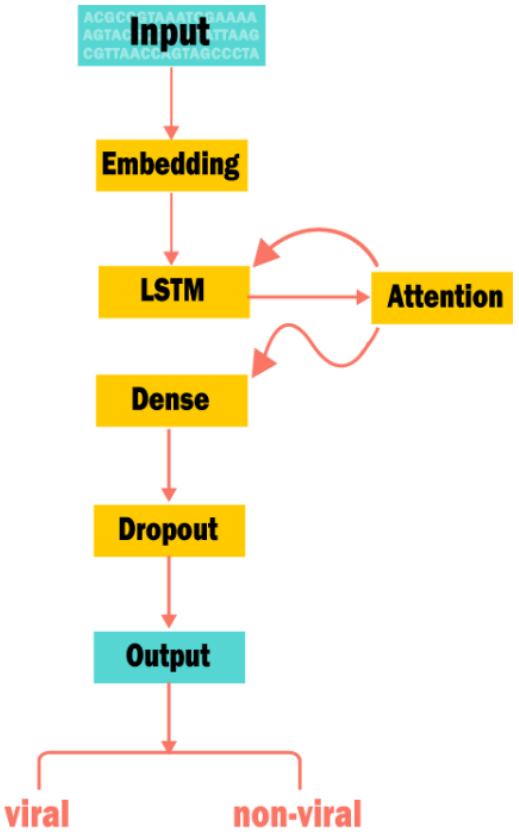
We ran other experiments with the same parameters to check the ability of our model with other configuration, so we changed different parameters separately such as embedding size to 256 neurons, the number of layers to 3 and the recurrent cell type to GRU instead of LSTM. We found a slightly less accuracy and ROC-AUC scores but the training time was much more than the best parameters configuration as expected due to increasing number of neural network parameters.

#Layers	Embedding	Ngram	ROC-AUC	Accuracy
1	32	3	0.8	73.66
1	32	5	0.83	76.42
1	32	7	0.79	72.11
1	64	3	0.83	76.1
1	64	5	0.83	75.98
1	64	7	0.77	69.96
1	128	3	0.83	75.76
1	128	5	0.85	77.41
1	128	7	0.78	70.93
2	32	3	0.8	73.25
2	32	5	0.83	76.49
2	32	7	0.79	72.82
2	64	3	0.81	73.96
2	64	5	0.84	76.46
2	64	7	0.78	72.53
2	128	3	0.83	76.15
2	**128**	**5**	**0.85**	**77.9**
2	128	7	0.78	70.63

TABLE 3.1: Hyperparamters optimization results



(A) Embedding Layer



(B) Neural network model architecture

FIGURE 3.1: VirNet model

Chapter 4

Experimental results

4.1 Data Generation

4.1.1 Building training and testing dataset

We downloaded viruses, bacteria and archaea genomes from RefSeq database then we divided them randomly into a train and test genomes with 80% of total base pairs in training. Table 4.1 shows the number of genomes we used in training and testing. We processed all available viral genomes until Nov. 1, 2017 and a sample from prokaryotic genomes due to the huge number of available prokaryotic genomes. Then, we converted the viral genomes into non-overlapping fragments of different lengths n = {100, 500, 1000, 3000}. We generated an approximate number of non-overlapping fragments of prokaryotic genomes with the same lengths randomly as well. (Table 4.2). We balanced the data of both classes using random under-sampling technique to avoid the bias to the majority class with the deep neural network. Figure 4.1 shows more details for data pipeline.

Genome	Train	Test	Total
Viruses	7686	1870	9556
Prokaryotes	143241	35543	178784

TABLE 4.1: The number of used genomes from RefSeq

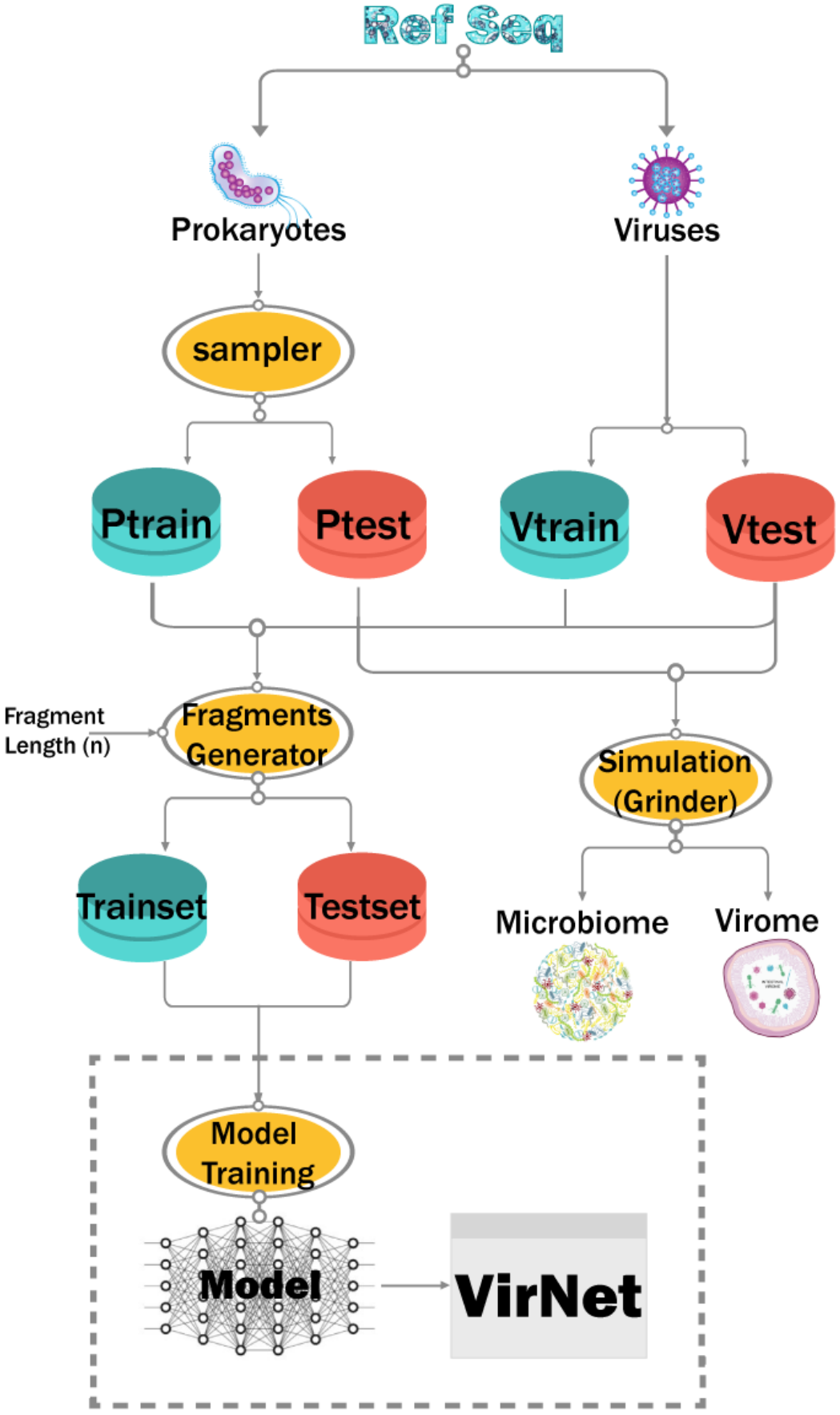


FIGURE 4.1: VirNet Data Pipeline

Fragment Length (N)	Train	Test
100 bp	2088863	527020
500 bp	420857	106168
1000 bp	212253	53528
3000 bp	73163	18425

TABLE 4.2: The number of fragments generated from viruses genomes

4.1.2 Generating simulated virome and microbiome

Grinder [19] is an open-source tool commonly used for generating a simulate amplicon and shotgun metagenomic datasets from reference genomes. We generated two metagenomic data of virome and microbiome of 1M reads and fragment length 100bp using Grinder with our reference test genomes to simulate shotgun metagenomic sequences in order to verify the ability of our tool to detect viral reads in metagenomic data instead of generated fragments from the reference genomes. The virome data has 75% of viral reads while microbiome has 25%. Moreover, we used Illumina error model indicated by mutation_dist poly4 3e-3 3.3e-8 and mutation ratio 91:9 (9 indels for each 91 substitution mutations) because for Illumina indel errors occur more often than substitution errors [20]. Table 4.3 shows simulated data statistics.

	Microbiome	Virome
Bacteria Length	75450367 bp	17551396 bp
Bacteria Genomes	1488	422
Bacterial reads	803742	176059
Viruses Length	25133078 bp	52609236 bp
Viruses Genomes	845	1870
Viral reads	196258	823941
Viral Ratio	25.00%	75.00%
Library coverage	0.994x	1.001x
Diversity (richness)	2302	2726

TABLE 4.3: Grinder Simulated Metagenome

4.1.3 Case Study: Real metagenomic data

We applied our tool to two real metagenomes as a case study

1. **454**: Subtropical freshwater microbial and viral metagenome (SRR648314).

2. **Illumina**: Lake Michigan virome (SRX995836).

Our tool is able to read not only fasta files and fastq files. Furthermore, it is able to deal with paired-end reads i.e. if one of the two pairs is identified as a virus, the other should be the same. If there are conflicts between the classifications of the two pairs; this pair could be denoted as ambiguous.

4.2 Results

4.2.1 Results for generated fragments

We tested VirNet on different lengths of fragments n= {100, 500, 1000, 3000} from our testing set of viruses and prokaryotes RefSeq genomes. Moreover, we compared the output results to VirFinder results on the same training and testing data. VirNet predictions outperformed VirFinder for fragments with length 500, 1000 and 3000 (Figure 4.2). The model reached to 82.82% of accuracy whereas VirFinder tool obtained 75.61%. Moreover, VirFinder can perdict the short fragments with length 100. Figures 4.3a and 4.3b shows ROC-AUC curves of both tools on the testing set. Table 4.4 shows the comparison between both tools in terms of accuracy, average precision and average recall.

Length(N)	Accuracy		Avg. Precision		Avg. Recall	
	VirNet	VirFinder	VirNet	VirFinder	VirNet	VirFinder
100	71.29%	63.9%	0.72	0.64	0.71	0.64
500	82.82%	75.61%	0.83	0.76	0.83	0.76
1000	86.82%	80.28%	0.87	0.82	0.87	0.80
3000	90.10%	87.11%	0.91	0.88	0.90	0.87

TABLE 4.4: Comparison on fragments test-set

4.2.2 Results for simulated metagenomes

As mentioned before, we tested VirNet on a simulated metagenomes of 100 bp and we found that VirNet performed much better than VirFinder. VirNet shows accuracy is 71.3% on the virome data and 72.14% on the microbiome data while VirFinder is 62.77% on the virome data and 64.49% on the microbiome data Table 4.5). The ROC-AUC curves of both tools shows the difference between them (Figures 4.4a and 4.4b).

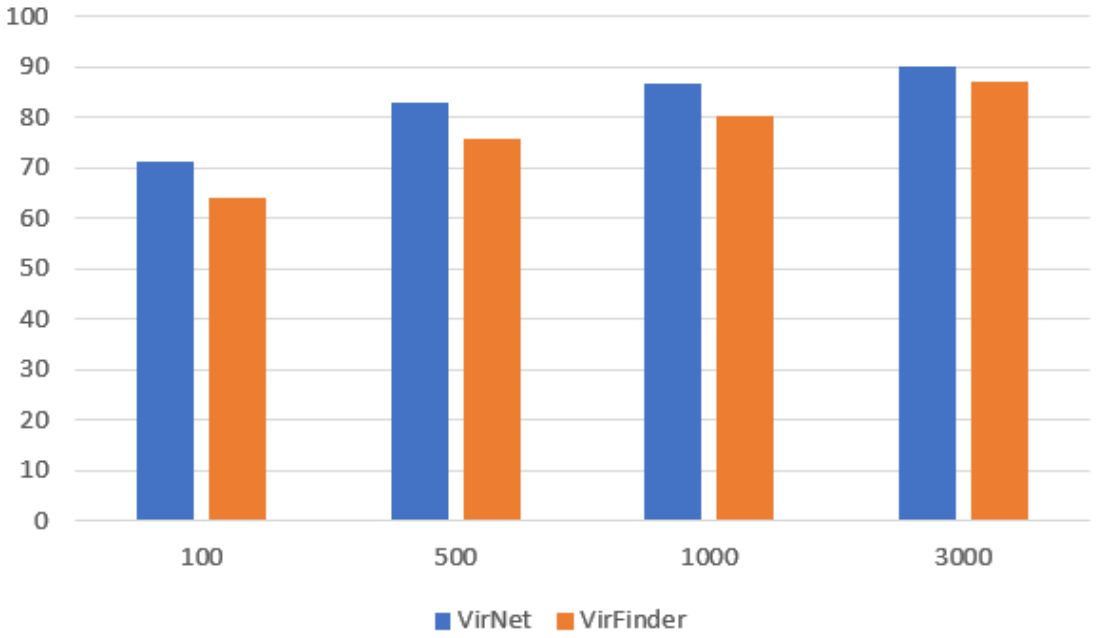


FIGURE 4.2: VirNet vs VirFinder Accuracy

Metagenome	Accuracy		Avg. Precision		Avg. Recall	
	VirNet	VirFinder	VirNet	VirFinder	VirNet	VirFinder
Virome	71.3%	62.77%	0.71	0.63	0.72	0.62
Microbiome	72.14%	64.49%	0.73	0.65	0.73	0.64

TABLE 4.5: Simulated Metagenomes Results

	SRR648314	SRR1974517/1	SRR1974517/2
Viral	39447	579697	590869
Non Viral	20866	1055975	1044803
Total	60313	1635672	1635672

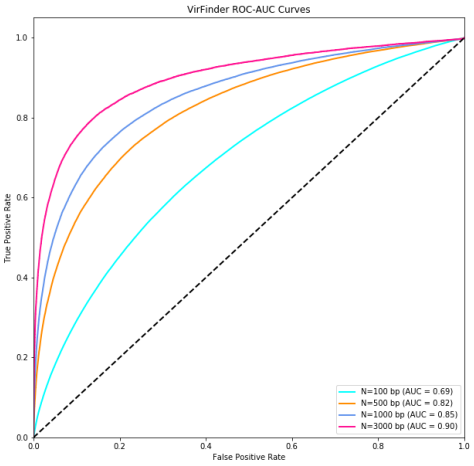
TABLE 4.6: VirNet results on real data

4.2.3 Results for real data

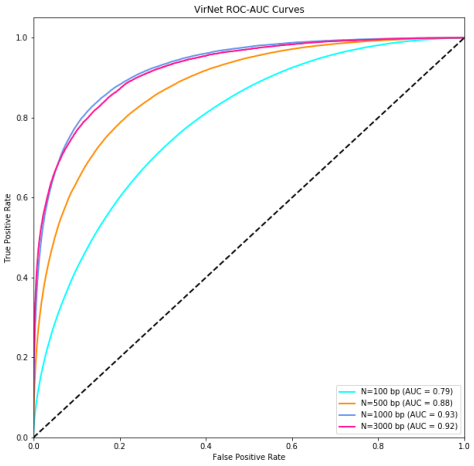
we applied the tool on two real data with accession numbers SRR648314, SRX995836_1 and SRX995836_2. Table 4.6 shows that our tool could able work on the real metagenomic data.

4.2.4 VirNet processing speed

Using GPUs is an advantage for our tool and make it very fast and scalable in processing massive amount of metagenomic reads simultaneously. The training process is around 5 hours with 2 million reads, while the prediction process is around 30 seconds on 1 million



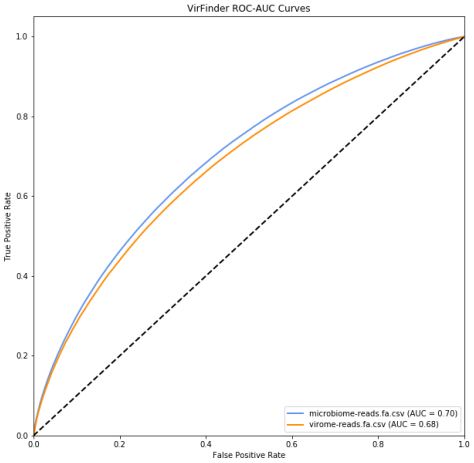
(A) VirFinder with generated fragments



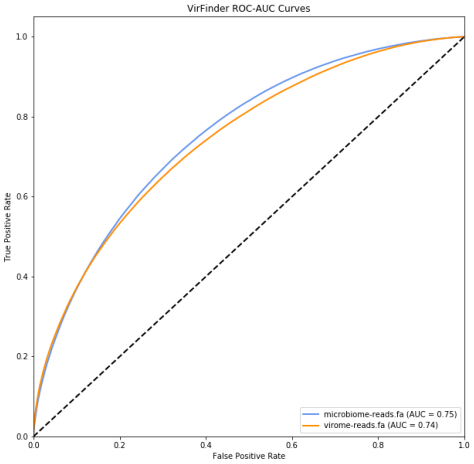
(B) VirNet with generated fragments

FIGURE 4.3: ROC-AUC curves on fragments

reads using Nvidia GeForce GTX 1080 Ti. On the other hand, VirFinder processing the reads on a single CPU less than VirNet by around 82 times. We can avoid that by using parallel CPU threads for VirFinder but in case you want to retrain it with new data, it will take a couple of days.



(A) VirFinder with simulated genomes



(B) VirNet with simulated genomes

FIGURE 4.4: ROC-AUC curves on simulated metagenomes

Chapter 5

Conclusion and Future Work

5.1 Summary

In our tool, there are no handmade features as the network will learn how to extract appropriate features of the raw data. It shows better accuracy as it is trained with the updated viral databases with a good statistical model. This helps us to generalize this model with all genomes and to make a generalized model for sequence classification. We are also able to identify the short viral sequences as LSTM learns from the dependences between the input sequence.

There is no evidence that these training prokaryotic genomes don't have a viral infection or not. Cleaning the training genomes might give us better accuracy but based on sampling and randomizing prokaryotic fragments, our training data may not contain proviruses.

For the trained deep learning model, using a sliding window over the input DNA sequence might improve our model, the only drawback of this technique is the slow training and inference time of input sequences, Also using an adaptive learning rate decaying over time steps during the training might improve the model performance, but will need more tuning over the input data.

5.2 Conclusion

This attentional neural deep learning network was able to achieve state of the art results on viral identification from high throughput sequences. Our approach is able to classify short fragments as well. Experimental results validate our approach for identification with an accuracy of more than 83%. According to these results, Our model would help us in understanding viruses in various microbial communities and discovering new species of viruses.

5.3 Future Work

we will work on this problem in the future ISA.

Appendix A

Appendix Title Here

Write your Appendix content here.

Bibliography

[1] Jacques Izard and Maria Rivera. *Metagenomics for Microbiology.* Academic Press, 2014.

[2] Jessica M Labonté, Brandon K Swan, Bonnie Poulos, Haiwei Luo, Sergey Koren, Steven J Hallam, Matthew B Sullivan, Tanja Woyke, K Eric Wommack, and Ramunas Stepanauskas. Single-cell genomics-based analysis of virus–host interactions in marine surface bacterioplankton. *The ISME journal*, 9(11):2386, 2015.

[3] Simon Roux, Steven J Hallam, Tanja Woyke, and Matthew B Sullivan. Viral dark matter and virus–host interactions resolved from publicly available microbial genomes. *Elife*, 4:e08490, 2015.

[4] Imane Allali, Jason W Arnold, Jeffrey Roach, Maria Belen Cadenas, Natasha Butz, Hosni M Hassan, Matthew Koci, Anne Ballou, Mary Mendoza, Rizwana Ali, et al. A comparison of sequencing platforms and bioinformatics pipelines for compositional analysis of the gut microbiome. *BMC microbiology*, 17(1):194, 2017.

[5] Samuel Minot, Rohini Sinha, Jun Chen, Hongzhe Li, Sue A Keilbaugh, Gary D Wu, James D Lewis, and Frederic D Bushman. The human gut virome: inter-individual variation and dynamic response to diet. *Genome research*, 2011.

[6] Jie Ren, Nathan A Ahlgren, Yang Young Lu, Jed A Fuhrman, and Fengzhu Sun. Virfinder: a novel k-mer based tool for identifying viral sequences from assembled metagenomic data. *Microbiome*, 5(1):69, 2017.

[7] Christof Angermueller, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. Deep learning for computational biology. *Molecular systems biology*, 12(7):878, 2016.

[8] Derrick E Fouts. Phage_finder: automated identification and classification of prophage regions in complete bacterial genome sequences. *Nucleic acids research*, 34(20):5839–5851, 2006.

[9] Gipsi Lima-Mendez, Jacques Van Helden, Ariane Toussaint, and Raphaël Leplae. Prophinder: a computational tool for prophage prediction in prokaryotic genomes. *Bioinformatics*, 24(6):863–865, 2008.

[10] You Zhou, Yongjie Liang, Karlene H Lynch, Jonathan J Dennis, and David S Wishart. Phast: a fast phage search tool. *Nucleic acids research*, 39(suppl_2): W347–W352, 2011.

[11] Sajia Akhter, Ramy K Aziz, and Robert A Edwards. Phispy: a novel algorithm for finding prophages in bacterial genomes that combines similarity-and composition-based strategies. *Nucleic acids research*, 40(16):e126–e126, 2012.

[12] Simon Roux, Francois Enault, Bonnie L Hurwitz, and Matthew B Sullivan. Virsorter: mining viral signal from microbial genomic data. *PeerJ*, 3:e985, 2015.

[13] K Eric Wommack, Jaysheel Bhavsar, Shawn W Polson, Jing Chen, Michael Dumas, Sharath Srinivasiah, Megan Furman, Sanchita Jamindar, and Daniel J Nasko. Virome: a standard operating procedure for analysis of viral metagenome sequences. *Standards in genomic sciences*, 6(3):421, 2012.

[14] Simon Roux, Michaël Faubladier, Antoine Mahul, Nils Paulhe, Aurélien Bernard, Didier Debroas, and François Enault. Metavir: a web server dedicated to virome analysis. *Bioinformatics*, 27(21):3074–3075, 2011.

[15] Benjamin Buchfink, Chao Xie, and Daniel H Huson. Fast and sensitive protein alignment using diamond. *Nature methods*, 12(1):59, 2014.

[16] Daehwan Kim, Li Song, Florian P Breitwieser, and Steven L Salzberg. Centrifuge: rapid and sensitive classification of metagenomic sequences. *Genome research*, 2016.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[18] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[19] Florent E Angly, Dana Willner, Forest Rohwer, Philip Hugenholtz, and Gene W Tyson. Grinder: a versatile amplicon and shotgun sequence simulator. *Nucleic acids research*, 40(12):e94–e94, 2012.

[20] David Laehnemann, Arndt Borkhardt, and Alice Carolyn McHardy. Denoising dna deep sequencing data—high-throughput sequencing errors and their correction. *Briefings in bioinformatics*, 17(1):154–179, 2015.