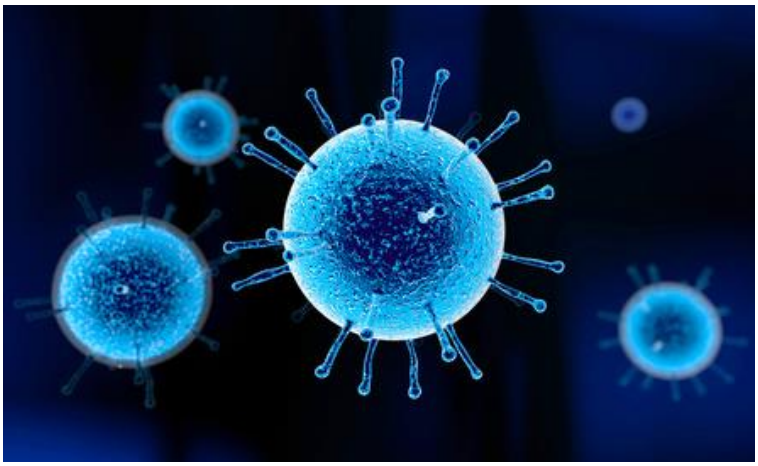
Internship Report

Do pressures from host immune systems lead to distinctive mutation patterns in the evolutionary history of viruses?



Author:
Xianli Li
Internship tutor:
Adrian Shepherd
School tutor:
Sylvain Viguier
Date**:**
03/05/2021 – 15/08/2021

Content

Acknowledgements

First of all, I would like to express my appreciation to my internship tutor, Adrian Shepherd, who has guided me through this project and this report, also has answered all my questions with patience.

Furthermore, I would like to thank my tutor of ESIEE Paris, Sylvain Viguier, who helped arrange meetings and follow my internship process.

Introduction

Under the current COVID-19 pandemic, we have come across an unexpected level of illness and death, and the global economic and social disruption is equally catastrophic. What's worse, like all viruses, the virus that causes COVID-19 -- SARS-CoV-2 changes over time. Most mutations are synonymous which don't change virus' proteins. However, some changes are non-synonymous, which means they can change the binding with lymphocytes to the virus then cause a viral escape, increase its spreading speed or the associated disease severity, eventually the effectiveness of vaccines.[1]

As one of the most effective measures to control the spread of this pandemic, scientists have confirmed that the effectiveness of COVID-19 vaccines is decreasing while facing one of the most contagious variants – Delta, which is due to the rapid viral evolution under pressures from host immune systems. And with the high level of transmission, mutated viruses are likely to continue to escape.[2] Therefore, this process has drawn our attention and brought out the principal question of this internship: **do pressures from host immune systems lead to distinctive mutation patterns in the evolutionary history of viruses?**

When a virus enters a new host species, for instance, when SARS-CoV-2 enters the human population, it undergoes host adaptation. This process can be studied using various methods for detecting evolutionary pressures on proteins, such as calculating the ratio between the number of non-synonymous and synonymous mutations. However, such approaches are typically poor at detecting the evolutionary pressure exerted by host immune systems. One of the reasons is that such pressure tends to be both local and transient since the focus of host immunity moves on to a different target. Moreover, host immune pressures come in many forms. Thus, instead of focusing on specific events in specific viruses, we decided to employ mathematical and computational methods to detect such patterns and to do so, without doubt, clustering algorithms are required. Since it only has been two years that COVID-19 appeared, it would be tricky to study its evolution over time. For this reason, we decided to choose influenza A, more specifically the subtype H1N1, for its significantly massive database through almost a hundred years, which allows us to improve the algorithm generalization.

In this report, I will present the university and the biology department where I did this internship. Also, the theoretical background knowledge in biology will be detailed and illustrated with figures, along with concrete implementation in Python. The third section will be mainly about the final results of the algorithm implementation together with its evaluation. The following part will be about methodologies applied during the internship, difficulties encountered and choices made. Finally, the conclusion will resume the whole project, what this internship has taught me and how to improve the results potentially in the future.

1. Presentation of the university

1.1. History

As a university with a long history, in 1823, “London mechanics’ Institute” was founded by Sir George Birkbeck. Several years later, the institute changed its structure to the University of London. After being through the First and the Second World Wars, Birkbeck never stopped teaching. Nowadays, Birkbeck is a public research university with nineteen academic departments in scientific area like biological science and computer science, but also in humanities such as history of art and cultural and languages, it is also well-known for its economics department.[3]

1.2. Department of Biological Science

In 2009, the Department of Biological Science was formed from two Schools, one of which — the School of Crystallography — had an international reputation for its structural biology research. The current Head of Department is Professor Carolyn Moores, who studies the cytoskeleton. [4]

2. Learning mutational patterns in epitopes using clustering

2.1. Background knowledge

Before getting started, we should first get familiar with basic biological background knowledge. More precisely, how the human immune system responds to viral invasion and the relevant research inspired us to commence this project.

2.1.1. Immune response

An immune response refers to a series of reactions that occurs in an organism to defend itself from the invasion from pathogens. It aims to fight against invasions from pathogens such as viruses, bacteria and parasites. Without the immune system, such pathogen invasion could cause serious health problems even endanger one's life. There are two main strategies of the immune response, the innate one and the adaptive one. These two collaborate to protect their organism from pathogens.

The innate immune system is the organism's first line of defence, which is non-specific and quick to all types of pathogens. However, the adaptive one is relatively slower since it protects one from specific pathogens.[5] Here is a graph demonstrating how the immune system responds to a viral invasion through the innate and the adaptive branches:

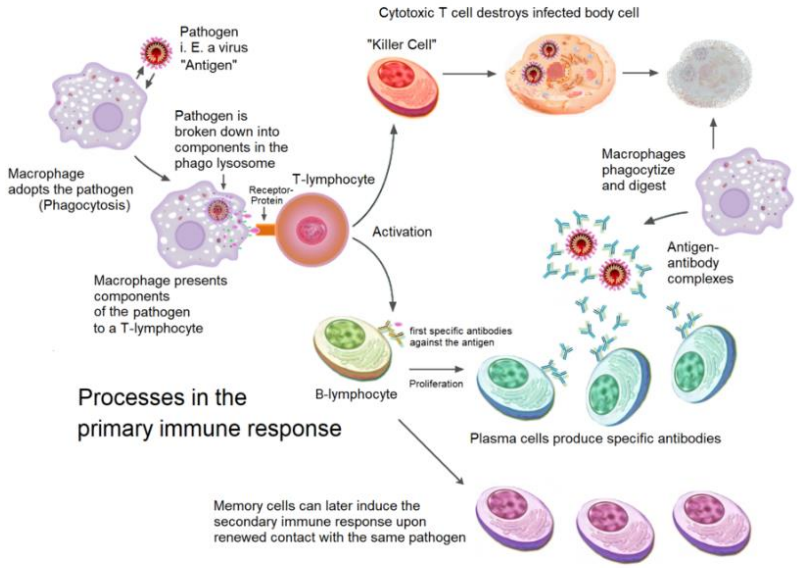


Figure 1 The immune response to a virus, the cytotoxic T cell response, shown here to the top right, is the main focus of this research[6]

As we can see from figure 1 above, when the virus invades an organism for the first time, it parasitizes the host cell to produce new proteins and replicate itself. Duplicated viruses get released once breaking through the host cell. These new viruses will continue to infect other healthy cells. Meanwhile, some of these pathogens are captured and engulfed by an antigen-presenting cell (APC). The APC will break them into peptides, which will be moved to the surface of APC to be recognized by activated T-lymphocyte cells. This process is named antigen presentation. Once activated, the T-lymphocyte cell generates lymphokines to signal B-lymphocyte and itself to proliferate and differentiate rapidly. Some of them stop the differentiation earlier and become memory T cells. The others continue to differentiate into many subtypes. One of the subtypes is called cytotoxic T cells (CD8+ T cells, killer T cells), a subtype of T cells that kill directly infected cells. On the other hand, when B-lymphocyte gets activated, as T-lymphocyte cell, it also divides and differentiates rapidly into plasma cells and memory B cells. The plasma cells generate a large number of antibodies then they release them into our circulatory system. These antibodies are keen to recognize and bind with the antigens that they have encountered. By doing so, antigens like viruses will lose the capacity of invading and infecting other cells.[7]

As we mentioned many times MHC molecules, it is essential to understand their functionality before proceeding. MHC stands for major histocompatibility complex, which is a large linkage group of DNA containing polymorphic genes. For humans, we call this complex Human Leukocyte Antigen (HLA). These genes code for proteins located on almost all nucleated cells to present pathogens peptides to killer T cells. They are called MHC molecules, and they play a crucial role in the adaptive immune system.[8] For human we call MHC as HLA. There are mainly three classes of MHC: MHC class I encodes peptide-binding proteins, which exist in all nucleated cells. They present epitopes to CD8+ T cells that detect the presence of antigens and destroy infected cells. However, we noticed that killer T cells only detect the epitopes when they can first recognize the bound MHC molecules. This phenomenon is called MHC restriction. MHC I molecules are the main focus of this research. Class II also encodes peptide-binding proteins. Most of them are located on the surface of APC to present the presence of antigen then activate T- lymphocyte. MHC class III encodes other immune proteins. Class II and III are irrelevant to this project.

2.1.2. Aim of the project

Once we get well-acquainted with how immune response works, we can take a second look at the main problem of this project: do pressures from host immune systems lead to distinctive mutation patterns in the evolutionary history of viruses? We know that immune systems respond in two ways, cell-mediated immunity, which eliminates infected cells by CD8+ T cells. And humoral immunity, which works by the specific combination between antibodies and antigens. Antibodies are usually large proteins that interact with antigen's epitope, a partial peptide of an antigen that killer T cells can recognize during immune response.[9] Like all proteins, antibodies are folded peptides in a specific 3D structure. Therefore, it is complicated to determine the individual amino acids residues or groups of residues that bind to an antigen's epitope. However, contrary to antibodies, CD8+ T cells recognize specific antigens by their T-cell receptors (TCR). When a cell gets infected by the influenza A virus, part of the viral peptide is bound to MHC class I molecules on the surface. The CD8+ T cell will recognize this epitope with its TCR and destroy the cell. Therefore, to study host immune pressure

on viral evolution, cell-mediated immunity is a better choice than humoral immunity in comparison. To conclude, this project aims to **analyse the pressure from cell-mediated immunity by focusing on the H1N1 virus' epitopes which CD8+ T cells can recognize through known MHC molecules**. As we mentioned before, CD8+ T cells recognize and destroy infected cells thanks to MHC molecules. But with MHC restriction, killer T cells only respond to limited MHC molecules, so before proceeding, we should consider this restriction by focusing on one of the numerous MHC alleles.

2.1.3. Relevant research

In one of the past researches in influenza A, we have seen mutational patterns by eye, figuring below:

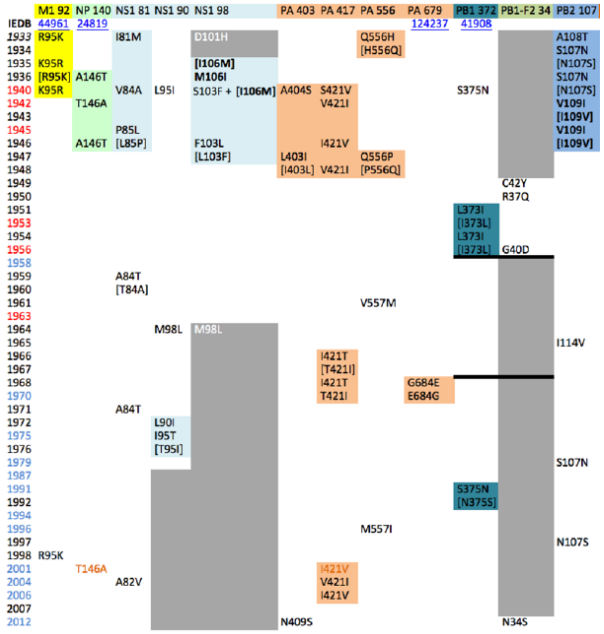


*Figure 2 HLA-A*01:01 9-mer epitope mutation activity over time, only mutations are marked in this figure, the columns are the 9-mer epitopes, rows represent years with mutations occurred[10]*

Here is a figure about substitutions arising within the internal proteins of influenza A H1N1 for individual 9-mer epitopes associated with the common allele HLA-A*01:01 in a given year. A substitution is a type of DNA replication error that places the wrong nucleotide or sequence of nucleotides in the wrong position, which results in non-synonymous mutations that change the amino acid type.[11] 9-mer refers to the length of the MHC class I groove, and HLA-A*01:01 is one of the most common human MHC alleles. In short, here we present mutations of 9-mer epitopes as length occurred in the history of H1N1, which bind with HLA-A*01:01 molecules. Every column is a 9-mer epitope represented by its first amino acid residue, and every row is a year with recorded H1N1 data. When substitution of a given epitope (column) arises in a given year (raw), we mark its position and its new amino acid type in this graph. For example, in the column PA417, the first mutation S421V means that at position 421, the original residue S mutated into V. What we observed is, for instance, still the column PA417, firstly there are mutations several years in a row, from 1940 to 1948, after that, no substitution arose in this epitope. But from 1966 to 1970, this epitope mutated again every year, then no mutation at all. This particular rhythm has drawn our attention, so we can't help but ask

questions: **is this mutation rhythm caused by pressure from the immune system? Can we also detect similar patterns with other epitopes and other MHC alleles?** To answer these questions, we thought about employ one of the machine learning methods since the traditional ones are typically poor at detecting the evolutionary pressures exerted by host immune systems, for example, the K_a/K_s ratio. Given that we aim to analyse significant unknown patterns instead of studying the existing ones, unsupervised learning such as clustering appears to be an appropriate choice.

2.2. Clustering implementation

As we explained earlier, clustering could be one of the solutions to our problem by gathering similar elements. Here is its implementation in detail.

2.2.1. Creation of the database

As we mentioned in the introduction, due to the lack of data and the short evolutionary history of COVID-19, we decided to choose influenza A, more accurately its subtype H1N1. Fortunately, Influenza Research Database (IRD) provides us with bioinformatics support for influenza virus research, especially its massive data from various sources and very competent research function. We can easily download data in categories like sequences and strains, immune epitope data or even 3D protein structure.[12]

To create our proper database, through the IRD website, we can filter and download all information in need. The Influenza A virus contains eight RNA segments that code for ten to fourteen proteins. The exact number depends on the strain. Here are the several basic proteins:

- Polymerase basic protein 2 (PB2 gene)
- RNA-directed RNA polymerase catalytic subunit and PB1-F2 protein (PB1 gene)
- Polymerase acidic protein and PA-X protein (PA gene)
- Hemagglutinin (HA gene), along with neuraminidase these two types protein decide the subtype of influenza A
- Neuraminidase (NA gene)
- Nucleoprotein (NP gene)
- Matrix proteins, M1 and M2 (M gene)
- Non-structural proteins, NS1 and NEP (NS gene)[13]

In our case, we decided to choose NS1 protein for that according to previous research, NS1 protein appears to be the most promising one which is associated with known T cell epitopes.

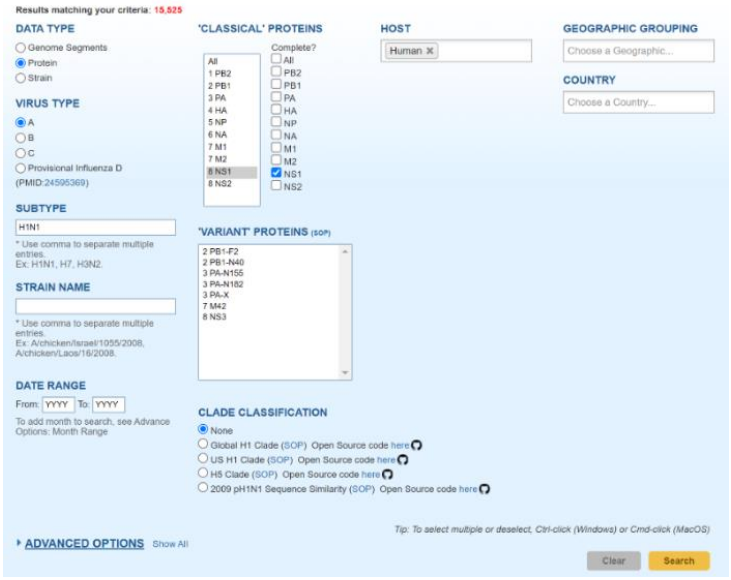


Figure 3 the IRD page to filter proteins as our need

Here is the IRD page. According to our need, we chose the option "protein" as data type since we aim to study influenza A virus epitopes, A stands for virus type, "H1N1" as subtype, "human host" and also "NS1 protein". By clicking on the button “search”, all filtered results show with the option of download as below:

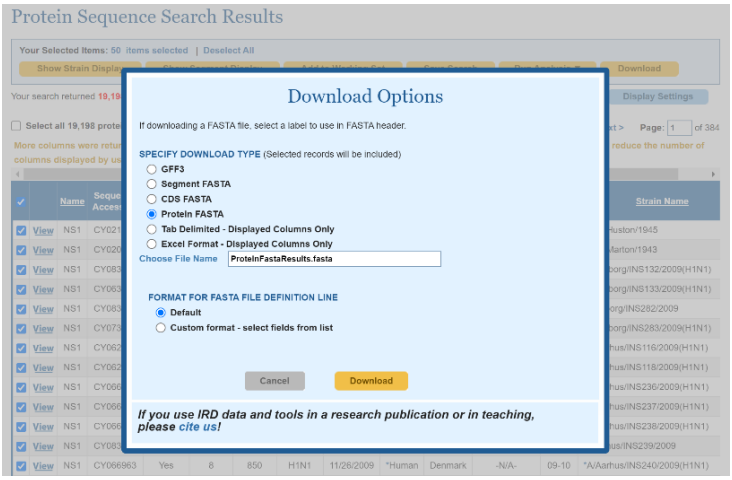


Figure 4 search results with several download choices

Then we choose “protein FASTA” as the file format to download. FASTA is a text-based format to represent nucleotide and protein sequences. Both nucleotides and amino acids types are represented by single-letter codes.[14] below here is an example of a protein’s information:

```
>gb:CY021713|ncbiId:ABP49332.1|UniProtKB:A4U6V8|Organism:Influenza A virus (A/AA/Huston/1945(H1N1))|
MDPNTVSSFQVDCFLWHVRKRVADQELGDAPFLDRLRRDQKSLRGRGSTLGLNIETATRVGKQIVERILK
EESDEALKMTMASALASRYLTDMTIEEMSRDWFMLMPKQKVAGPLCIRMDQAIMDKSIILKANFSVIFGR
LETLILLRAFTEEGAIVGEISPLPSLPGHTNEDVKNAIGVLIGGLEWNDNTVRVSKTLQRFAWRSSNENG
RPPLTPKQKRKMARTIRSEV
```

Figure 5 example of protein information in FASTA format

As we can see in figure 5, the headline begins with a '>' symbol. It contains a unique identifier for the sequence and some supplementary information. For example, in this figure, both the organism of the protein and the NCBI ID are presented. Following the headline, we can see the protein sequence, represented by letters of amino acid types.[14]

2.2.2. Data pre-processing

Previously, we have obtained the data in FASTA format. Even though this format possesses numerous advantages, it still has many restrictions for machine learning purposes. The sequence example in figure 5 contains some additional information: NCBI ID, UniProtKB and the organism of the protein. But this information is all in the same line. To run a machine learning algorithm, we prefer to process these data with separate columns. This disadvantage of FASTA files leads to data pre-processing with "pandas", a well-known library for Python users to process data and run analysis.[15] We first pre-process the data from the FASTA file that we downloaded, then store it into pandas DataFrame to simply the embedding method afterwards. Here is a glance of chronologically sorted data:

UniProtKB	Organism	Strain Name	Protein Name	Gene Symbol	Segment	Subtype	Host	Year	sequence
Q99AU3	Influenza A virus (A/Brevig Mission/1/1918(H1N1))	A/Brevig Mission/1/1918	NS1 Non-structural protein 1	NS1	8	H1N1	Human	1918	MDSNTVSSFQVDCFLWHVRKRFADQELGDAPFLDRLRRDQKSLRGR...
Q20MH3	Influenza A virus (A/Wilson-Smith/1933(H1N1))	A/Wilson-Smith/1933	NS1 Non-structural protein 1	NS1	8	H1N1	Human	1933	MDPNTVSSFQVDCFLWHVRKRVADQELGDAPFLDRLRRDQKSLRGR...
0A0C6DX30	Influenza A virus (A/NWS/1933(H1N1))	A/NWS/1933	NS1 Non-structural protein 1	NS1	8	H1N1	Human	1933	MDPNTVSSFQVDCFLWHVRKRVADQELGDSPFLDRLRRDQKSLRGR...
Q67249	Influenza A virus (A/NWS/1933(H1N1))	A/NWS/1933	NS1 Non-structural protein 1	NS1	8	H1N1	Human	1933	MDPNTVSSFQVDCFLWHVRKRVADQELGDSPFLDRLRRDQKSLRGR...

Figure 6 the example of NS1 proteins sequences stored in a DataFrame, sorted chronologically by year

As we said earlier, our goal is to study the pressure from cell-mediated immunity by focusing on the H1N1's epitopes which CD8+ cells recognize. Our strategy is to cut NS1 protein sequences into 9-mer segments, which is the length of peptide binders, then study the evolution of each 9-mer epitope in time and cluster the similar ones together. Since CD8+ T cells cannot bind with the whole protein, we need to predict the binding epitopes. With the NetMHCpan-4.1 server, we can employ many powerful tools powered by artificial neural networks (ANN) to predict peptide binders to any MHC molecules.[16] By clustering these predicted binders, we can prove if there are immune system driven mutational patterns. Here is an extraction of 9-mer segments of NS1 protein in the history of the H1N1 virus:

year	0	1	2	3	4	5	6	7	8	9	...	21
1918	MDSNTVSSF	DSNTVSSFQ	SNTVSSFQV	NTVSSFQVD	TVSSFQVDC	VSSFQVDCF	SSFQVDCFL	SFQVDCFLW	FQVDCFLWH	QVDCFLWHV	...	PLPPKQKR
1933	MDPNTVSSF	DPNTVSSFQ	PNTVSSFQV	NTVSSFQVD	TVSSFQVDC	VSSFQVDCF	SSFQVDCFL	SFQVDCFLW	FQVDCFLWH	QVDCFLWHV	...	PLTPKQKR
1934	MDPNTVSSF	DPNTVSSFQ	PNTVSSFQV	NTVSSFQVD	TVSSFQVDC	VSSFQVDCF	SSFQVDCFL	SFQVDCFLW	FQVDCFLWH	QVDCFLWHV	...	PLTPKQKR
1935	MDPNTVSSF	DPNTVSSFQ	PNTVSSFQV	NTVSSFQVD	TVSSFQVDC	VSSFQVDCF	SSFQVDCFL	SFQVDCFLW	FQVDCFLWH	QVDCFLWHV	...	PLTPKQKR
1936	MDPNTVSSF	DPNTVSSFQ	PNTVSSFQV	NTVSSFQVD	TVSSFQVDC	VSSFQVDCF	SSFQVDCFL	SFQVDCFLW	FQVDCFLWH	QVDCFLWHV	...	PLTPKQKR

Figure 7 example of 9-mer segments of NS1 protein in the history of H1N1, we generate them for the embedding purpose

As demonstrated in figure 7, we sample residues by overlapping 9-mers such as positions 0 to 8, 1 to 9, 2 to 10. These 9-mers are columns in order of the years. This approach can give us more data to analyse. Vertically we have obtained the evolution of every 9-mer in each column, and the rows refer to corresponding years.

2.2.3. Sequence embedding

As we know that sequence modelling has never been easy, which is due to its non-structural properties. Traditionally, during Natural Language Processing (NLP) tasks, texts tend to have words in a specific order to form sentences with meanings and semantics. Protein sequences are arbitrary strings with an alphabet, which results in the difficulty of vectorization since the traditional embedding methods become incompetent.[17] And one of the biggest challenges of this project is to devise ways of encoding information about how individual residues, and groups of neighbouring residues, mutate over time. Therefore, **we should encode the mutation rhythm and position rather than focusing on the specific amino acid type and the actual year.**
To solve this problem, we introduce the concept of embedding, which is a representation of texts in a low-dimensional space while keeping the same meanings. It certainly makes the machine learning tasks easier to conduct by reducing the input dimensions, and ideally, embeddings can also preserve the semantics of the input texts.[18] The word embedding method is widely used in the NLP area to understand natural language, which can also help us understand the sequential data. As mentioned above, traditional embedding methods tend to be incompetent facing sequences since they are arbitrary strings rather than semantic sentences. After thorough research and countless tests, we have found a library called Sequence Graph Transform (SGT) – a sequence embedding tool for clustering and classification purposes. It embeds both long- and short-term patterns of the sequential dataset in the Euclidean space. With SGT, we can easily vectorize our protein sequences without increasing the computation.[17] Here are the embeddings in two dimensions of each 9-mer segment above:

	0	1	2	3	4	5	6	7	8	9	...	212	213
1918	[4.5424166, 19.299679]	[4.7259555, 19.183111]	[4.9907546, 18.975039]	[5.3595195, 18.868305]	[5.6530924, 18.717121]	[5.9054112, 18.585846]	[6.156302, 18.445017]	[6.246691, 18.405075]	[6.341852, 18.351276]	[6.4342556, 18.304033]	...	[-11.587673, 12.235334]	[-10.768061, 13.50531]
1933	[4.5424166, 19.299679]	[4.7259555, 19.183111]	[4.9907546, 18.975039]	[5.3595195, 18.868305]	[5.6530924, 18.717121]	[5.9054112, 18.585846]	[6.156302, 18.445017]	[6.246691, 18.405075]	[6.341852, 18.351276]	[6.4342556, 18.304033]	...	[-11.587673, 12.235334]	[-10.768061, 13.50531]
1934	[4.5424166, 19.299679]	[4.7259555, 19.183111]	[4.9907546, 18.975039]	[5.3595195, 18.868305]	[5.6530924, 18.717121]	[5.9054112, 18.585846]	[6.156302, 18.445017]	[6.246691, 18.405075]	[6.341852, 18.351276]	[6.4342556, 18.304033]	...	[-11.587673, 12.235334]	[-10.768061, 13.50531]
1935	[4.5424166, 19.299679]	[4.7259555, 19.183111]	[4.9907546, 18.975039]	[5.3595195, 18.868305]	[5.6530924, 18.717121]	[5.9054112, 18.585846]	[6.156302, 18.445017]	[6.246691, 18.405075]	[6.341852, 18.351276]	[6.4342556, 18.304033]	...	[-11.587673, 12.235334]	[-10.768061, 13.50531]
1936	[4.5424166, 19.299679]	[4.7259555, 19.183111]	[4.9907546, 18.975039]	[5.3595195, 18.868305]	[5.6530924, 18.717121]	[5.9054112, 18.585846]	[6.156302, 18.445017]	[6.246691, 18.405075]	[6.341852, 18.351276]	[6.4342556, 18.304033]	...	[-11.587673, 12.235334]	[-10.768061, 13.50531]

Figure 8 embeddings of each 9-mer segment based on 9-mer segments that we generated previously

2.2.4. Time series representation

Now we are facing another big issue: **how to study the evolution of those 9-mers from their embeddings while blurring the specific mutation year and amino acid type**? Since the viral evolution refers to continuous mutations over time, we can consider it a time series, a sequence of chronologically sorted data[19]. With embeddings in Euclidean space, we can easily calculate the Euclidean distance between every two successive 9-mers embeddings. By replacing the embeddings of 9-mers with its distance with the next one, we can easily ignore its amino acid type. Below I present the evolution of 9-mers in the form of time series:

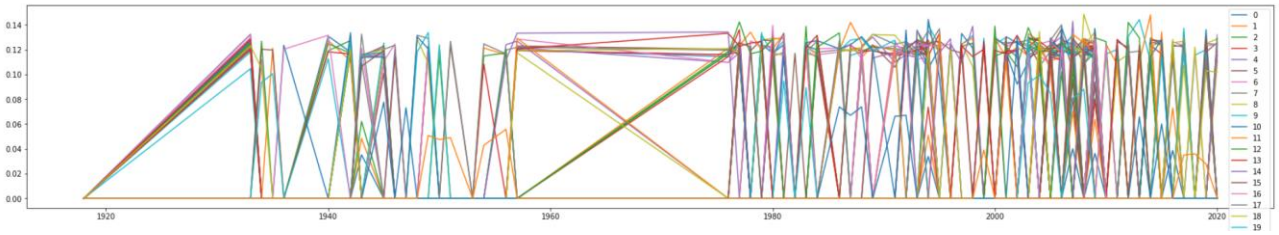


Figure 9 the evolution of 9-mers in the form of time series, each time series represents a 9-mer's evolution through 100 years, in which the value of each point is the Euclidean distance with the one from the previous year, calculated based on their embedding

When there is no mutation between two years, the distance equals 0. With this interpretation, the frequency of mutations is presented, and the position will be the value of each Euclidean distance. Based on this representation, we have transformed the simple clustering problem into a time series clustering problem.

2.2.5. K-means clustering

Since we have transformed the problem into a time series clustering problem, we employed a time series package called "tslearn", which provides machine learning tools for time series analysis. Their clustering API is keen to gather similar time series together, which allow us to capture mutational patterns over time.[20] Here is the output of K-means clustering using `tslearn.clustering.TimeSeriesKMeans` class:

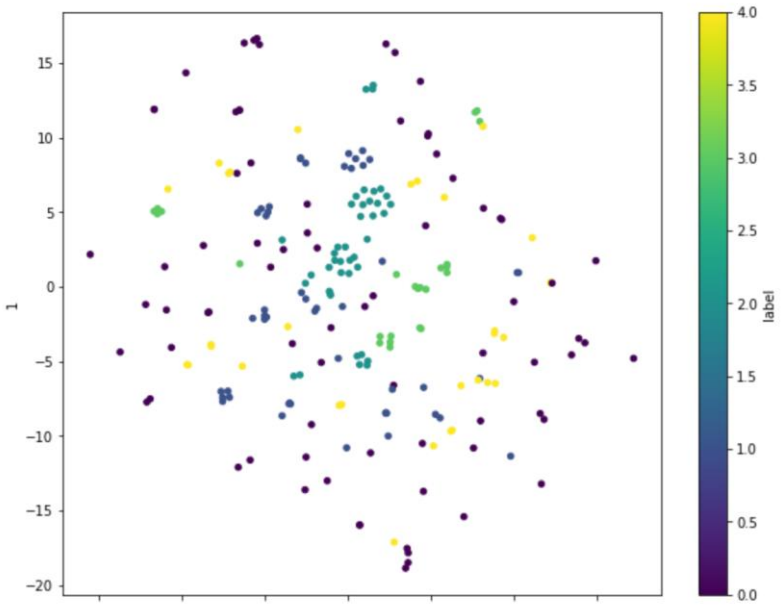


Figure 10 K-means clustering output, each point represents a time series, the points in the same colour belong to the same cluster

This graph shows the output of clustering from a global vision. We predefined the number of clusters as five, each point above represents a time series of the 9-mer, and clusters differ in their colour. As we can see, indeed the similar time series are gathered together, but they still disperse. This observation could be telling us that some of the clusters are poorly separated, or there is a lack of internal cohesion.

3. Results and evaluation

3.1. Results

Following the previous clustering output, we have clustered similar time series. But we still need to see if there are consistent high numbers of the same epitope in one or two clusters to prove that the immune system does have pressure upon viral evolution since they share the same mutational pattern.

3.1.1. Selection of binders

Since the number and position of epitopes mutate over time, we decided to choose one sequence every ten years to predict its 9-mer epitopes to HLA-A02:01, the most common class I HLA allele, using NeMHCpan. This server predicts both strong and weak binders (epitopes) like below:

```
weak_binders = {
    1918 : ["QVDCFLWHV", "TLGLDIETA", "ILKEESDEA", "MLMPKQKVA", "AIVGEISPL"],
    1947 : ["QVDCFLWHV", "TLGLNIETA", "ILKEESDEA", "MLMPKQKVA", "AIVGEISPL"],
    1977 : ["QVDCFLWHV", "TLGLNIETA", "ILKEESDEA", "MLMPKQKVA", "AIVGEISPL"],
    1988 : ["TLGLNIETA", "TLKMNIASV", "FMLMPRQKI", "AIVGEISPL"],
    1999 : ["TLGLNIETA", "TLKMNIASV", "FMLMPRQKI", "AIVGEISPL"],
    2009 : ["TLGLDIETA", "TLRMTIASV", "FMLMPRQKI", "MLMPRQKII", "IIGPLCVRL", "IVLKANFSV", "AIVGEISPL"],
    2019 : ["TLGLDIKTA", "TLRMAIASV", "FMLMPRQKI", "MLMPRQKII", "IIGPLCVRL", "TIVGEISPL"]
}
strong_binders = {
    1918 : ["ALKMTIASV", "FMLMPKQKV", "AIMDKNIIL", "IILKANFSV", "VIFDRLETL"],
    1947 : ["FLWHVRKRV", "FMLMPKQKV", "AIMDKSIIL", "IILKANFSV", "VIFDRLETL"],
    1977 : ["FLWHVRKQV", "FMLMPKQKV", "AIMDKNIIL", "IILKANFSV", "VIFDRLETL"],
    1988 : ["GLNIETATL", "AIMEKNIIL", "IILKANFSV", "VIFNRLETL", "IVGEISPLL"],
    1999 : ["GLNIETATL", "AIMEKNIIL", "IILKANFSV", "VIFNRLETL", "IVGEISPLL"],
    2009 : ["GLDIETATL", "AIMEKNIVL", "VIFNRLETL"],
    2019 : ["GLDIKTATL", "AVMDKNIVL", "IVLEANFSV", "VIFNRLETL"]
}
```

Figure 11 predicted weak and strong binders of 10 sampled sequences using NeMHCpan

3.1.2. Visualization of binders

The next thing to do is to visualize clusters and the position of binders in their clusters:

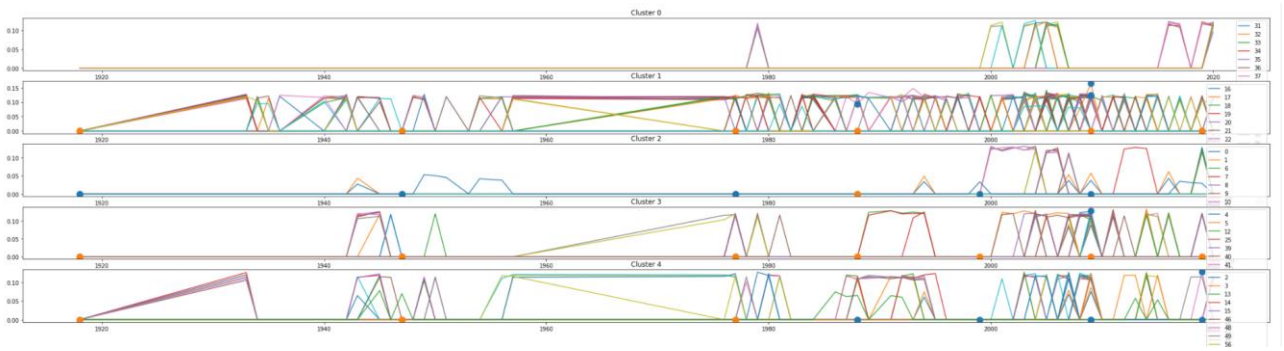


Figure 12 5 time series clusters and binders, weak binders mutate over time but the strong ones don't mutate

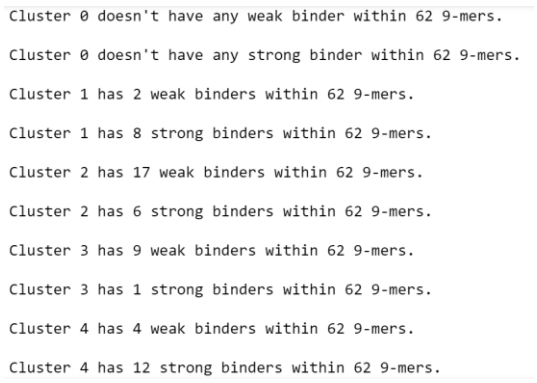


Figure 13 numbers of weak and strong binders in each cluster

Here is an example of a clustering output. In figure 12, weak binders are marked in blue and strong binders in orange, as we can see from these two graphs, even though the binders in similar shapes gathered together, like clusters 0 and 2. The binders appear to belong to four of five clusters, but the output of each run can be different from each other. Sometimes binders only appear in three of five clusters. This problem is due to one of the disadvantages of K-means clustering. It first selects randomly centroids, which are the initial value of each cluster. Then it optimizes the positions of these centroids with a certain number of iterations. When we have stabilized randomly selected centroids or the iterations are all finished, the algorithm stops.[21] In our case, some of the clusters are poorly separated or lack internal cohesion. Therefore, the outputs of clustering tend to be unstable. But the stochastic initiation is inevitable in the majority of clustering problems. To solve this problem, we have set the hyperparameter — the number of times the algorithm will run with different centroid seeds – like 20, the final result will be the best output among these 20 runs. We hoped such a method could stabilize the clustering outputs of each run, but unfortunately, the clustering output is still unstable. Our solution is to measure cluster consistency. We will verify if there is one or several groups of binders that always gather together during every run of clustering.

3.2. Evaluation

3.2.1. Intersections to remove noises

As we can see from figure 12, weak binders mutate over time, but strong binders never mutate. There could be many reasons to explain this phenomenon: it may be in the interests of a virus to mutate epitope A because it drives an effective immune response, but preserve epitope B since it disrupts an effective immune response. Whereas epitope C may be difficult to change because doing so undermines the function of the protein. In any case, to study mutational patterns, we shouldn't consider the immutable epitopes. Under this circumstance, we will only work on weak binders to evaluate the clustering output.

The strategy here is to run several times the clustering algorithm, store the clusters each time, then intersect the clusters with each other so that we can obtain a certain number of intersections. These intersections are groups of persisting binders. Here are the cluster intersections that we obtained after

twenty runs:

```
[{1988: 'TLKMNIASV', 2009: 'TLRMTIASV'},
 {1918: 'MLMPKQKVA',
  1947: 'MLMPKQKVA',
  1977: 'MLMPKQKVA',
  2009: 'MLMPRQKII',
  2019: 'MLMPRQKII'},
 {1988: 'FMLMPRQKI', 2009: 'FMLMPRQKI', 2019: 'FMLMPRQKI'},
 {1918: 'TLGLDIETA',
  1947: 'TLGLNIETA',
  1977: 'TLGLNIETA',
  1988: 'TLGLNIETA',
  1999: 'TLGLNIETA',
  2009: 'TLGLDIETA',
  2019: 'TLGLDIKTA'}]
```

Figure 14 common binders in each clustering, between each pair of braces there a group of binders that always gather together in every clustering

We have run twenty times to generate their intersections and to guarantee that they are the minimum ensembles. We have run several times this programme. Fortunately, the cluster intersections stay the same in each run. To visualize, we have run the programme for the second time and plotted 9-mer binders:

```
Cluster 0 has 3 binders:
{1988: 'FMLMPRQKI', 2009: 'FMLMPRQKI', 2019: 'FMLMPRQKI'}

Cluster 0 has 7 binders:
{1918: 'TLGLDIETA', 1947: 'TLGLNIETA', 1977: 'TLGLNIETA', 1988: 'TLGLNIETA', 1999: 'TLGLNIETA', 2009: 'TLGLDIETA', 2019: 'TLGLD
IKTA'}

Cluster 3 has 2 binders:
{1988: 'TLKMNIASV', 2009: 'TLRMTIASV'}

Cluster 3 has 5 binders:
{1918: 'MLMPKQKVA', 1947: 'MLMPKQKVA', 1977: 'MLMPKQKVA', 2009: 'MLMPRQKII', 2019: 'MLMPRQKII'}
```

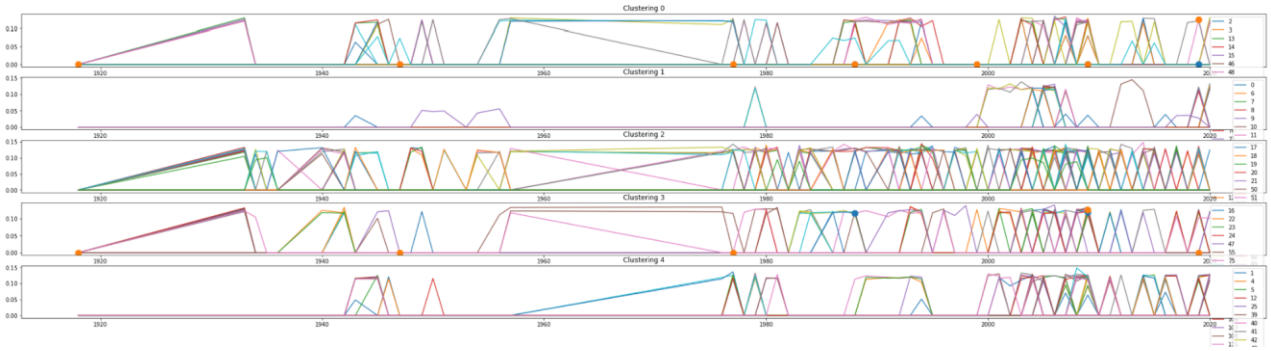


Figure 15 clustering with intersection output marked in orange and blue, 2 groups of binders are clustered in cluster 0, and the other 2 in cluster 3

In this example, there are two groups of binders in cluster 0 and cluster 3, and sometimes there are three of them in one cluster, but in any case, binders in the same group always gather together. Their existence proves in a certain way that despite the poor internal cohesion between the clusters, we still have successfully clustered persisting 9-mer binders, which means that their timeline profiles share characteristics that we have captured in the way we encode the data.

But on the other hand, we have sampled ten sequences then predicted ten weak and strong binders for each protein sequence. Among these 100 binders, half of them are strong binders that never mutate

during this research, and the clustering output on the other half, the weak binders, changes during every run. We suppose that there could be a lack of internal cohesion. After intersected the clusters between each run, indeed, we have successfully generated these intersections. But we only obtained 20 binders out of 50, which means only 20 out of 50 have a strong cohesion, and the rest of them are merely noises. Meanwhile, we have captured similar patterns between the 9-mers, and the time series in the same cluster share a mutational pattern in a certain way. But it is not obvious enough from the clustering figure that one specific mutational pattern could represent the gathered time series in the same cluster.

3.2.2. Evaluation result

To conclude, we should admit that from what we have seen, we can observe a sign of patterns from figure 15, since we can see some similarities inside of clusters, and we have captured persisting binders during each clustering. However, what we look forward to seeing is that all or nearly all of the variable predicted binders are consistently clustered together in very few clusters so that our supposition could be proved: while binding with epitopes, T cell receptors have exerted evolutionary pressure on the H1N1 virus. Unfortunately, in our case, the binders are too separated during the clustering. The reasons could be one of the following:

1. The clustering couldn’t generate a strong separation between predicted variable epitopes and other peptides, since the 9-mers with similar mutational patterns and clustered together, but the binders are not -- they spread almost in every cluster, and they could be clustered differently during each run
2. The NeMHCpan server is indeed powerful but not perfect. It is always possible that the wrong binders are predicted but the real binders are left out. Some non-binder peptides may share some of the adaptive properties of these epitopes. Besides, the temporal dynamics of "predicted variable epitopes” by NeMHCpan could be highly variable, so it is arguably unrealistic to expect them to resolve into a single cluster.
3. There could be more sophisticated data-encoding strategies that can cluster and capture more accurately the shared properties of "predicted variable epitopes”

Meanwhile, some other factors could influence the clustering output that we don't count as one of these reasons above since we have already optimized them. For instance, we don't consider changing the HLA allele type to increase the clustering ability since we have predicted binders of the most common one, HLA-A02:01 so that it is the most likely to discover a mutational pattern. We have also chosen the NS1 protein because, according to former research, NS1 protein is more promising to bind with T cell receptors. Besides these two possible factors, during the clustering, we have chosen K-means with Dynamix Time warping (DTW, a similarity measure) rather than Kernel K-means, because we focus on the shape of time series instead of their phase of changing.

In conclusion, with the current method of sequence embedding and time series clustering, we can detect some similar distinctive mutational patterns on certain 9-mers, but we cannot prove that such patterns are driven by the host immune system.

4. Analysis and methodology

4.1. Difficulties

During this internship, one of the difficulties I have encountered is the lack of a biology background. In high school, we have merely learned about the basic concept of the immune system, so in the beginning, just understanding the aim of this project was complicated for me. For this reason, I began this project by learning the technical terms and how the immune response works during a month.

During the implementation in Python, the most challenging part is, without a doubt, to encode information about how individual residues, and groups of neighbouring residues, mutate over time. Besides, since we focus on the frequency and positions of viral mutations, which means the specific years and mutated amino acid type don't matter, this constraint makes the implementation even harder. Inspired by the "pandas" method – `pandas.DataFrame.diff()`, which generates the difference between a DataFrame element and the previous one[22] , we could also calculate the difference of a sequence embedding with its previous one to identify the mutation. Since the embeddings are already in Euclidean space, this difference represents the position of the corresponding mutation. When there is no mutation, this difference equals zero. In the form of time series, we can analyse this frequency like in figure 9.

Another difficulty that we have encountered is that the clustering outputs change during each run. It makes the evaluation harder to proceed with since the clusters never gather the same time series. Hence, we did the intersection between clusters of each run to find the persisting ones, which we consider the final output of clustering.

4.2. Choices made

During data pre-processing, I hesitated to read and process the data with bioinformatics tools like "biopython", which provides many power methods. For instance, "biopython" has functions to read and write files in different formats like the FASTA format that we used in this research, also machine learning methods in bioinformatics like clustering. On the other side, "pandas" provides more data manipulation methods to process the dataset according to our needs. Finally, I have decided to combine these two powerful libraries by reading the FASTA file with "biopython" for its facility in processing bioinformatic files, then manipulating the dataset with "pandas" for its powerful tools to process our dataset.

Then the second choice that I have to make is to choose an embedding method that responds perfectly to our needs: embedding nearly fourteen thousand short sequences. Unlike the datasets used in traditional Natural Language Processing (NLP) tasks, they usually have words in specific order to form sentences with meanings and sentiments. Protein sequences are arbitrary strings with an alphabet, which results in the difficulty of vectorization since the traditional embedding methods

become incompetent. After trying most of the existing libraries, I have found the one which fits our need: the Sequence Class Definition (SGT) embedding library. It helps us to easily embed short- or long-term sequence features without increasing the computation. And more importantly, it is easy to understand its logic and employ even for beginners in bioinformatics.

After the sequence embedding, we have obtained an embedding matrix with nearly 400 dimensions. It can exponentially increase the computation during clustering. We need to reduce its dimensions between two methods: principal component analysis (PCA) and t-distributed stochastic neighbour embedding (t-SNE). Eventually, we chose t-SNE because the PCA method reduces the number of dimensions mathematically by using the correlation between the provided dimension then representing them with fewer variables while maintaining the information to the maximum. But the t-SNE analyses the raw dataset with probabilistic methods then represents it while using fewer dimensions by matching both distributions.[23] This advantage makes t-SNE more accurate and well suited for high-dimensional problems. [24]

Since we have transformed the problem into a time series clustering problem, we employed a time series package called "tslearn", which provides machine learning tools to analyses time series. Regarding our problem, it provides us with two main solutions: K-means with Dynamix Time warping (DTW, a similarity measure) and kernel-K-means. Different from K-means clustering based on the Euclidean distance metric, the DTW metric is sensitive to time shifts, which allows us to gather time series of similar shapes. Another solution to the previous problem with time shifts is kernel-K-means, inspired by DTW metric method. One of the significant differences compared with K-means is that clusters generated by kernel-K-means depend on the phase of time series rather than their shape. In short, kernel-K-means differ time series based on their difference in a given time, but K-means focus on the different shapes of time series. Here is an example of two different results of these two methods on the same dataset:

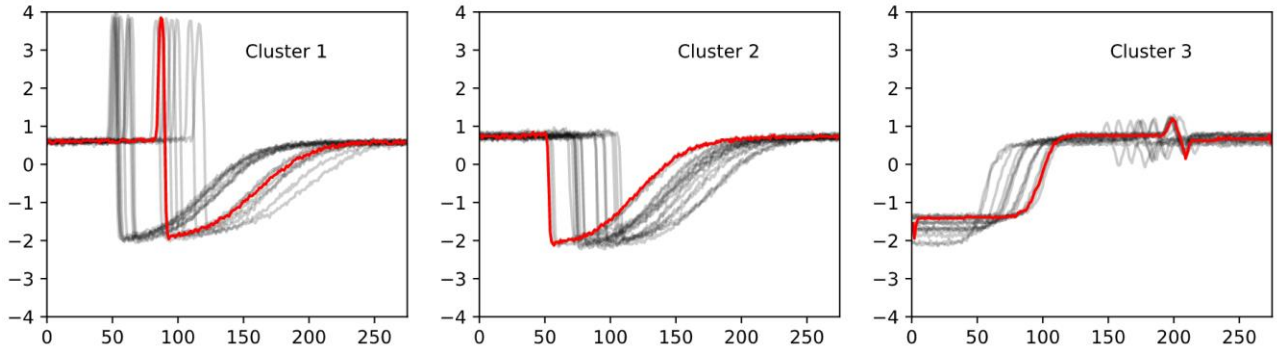


Figure 16 example of K-means clustering with DTW. As we can see, time series of the same cluster share the same shape. Their centroids are marked in red[20]

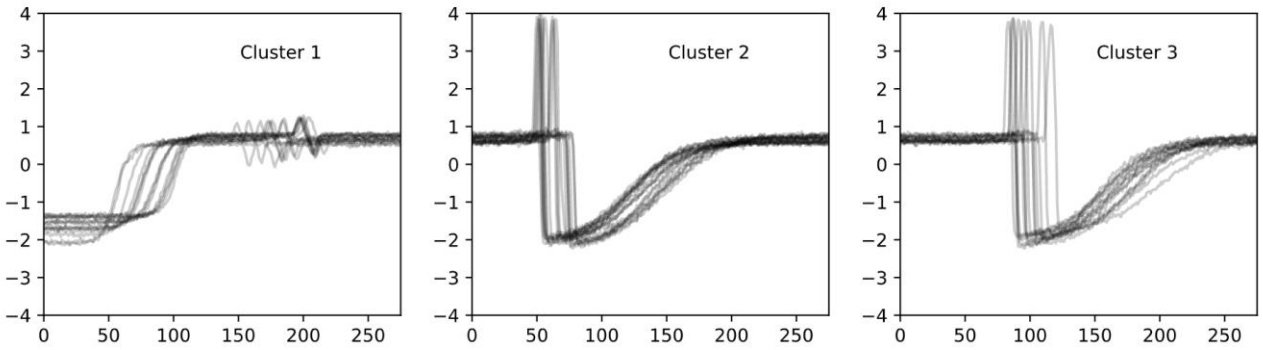


Figure 17 example of kernel-K-means clustering outputs. Compared with the previous clustering, kernel-K-Means clustered time series which share a similarity in the given phase[20]

Theoretically, to study a rhythm of viral evolution, we are more interested in the similar shape of time series than their similarity in the given phase since we focus on the frequency and the position of mutation while ignoring its specific year. However, the clustering outputs of these two methods on our dataset are very similar:

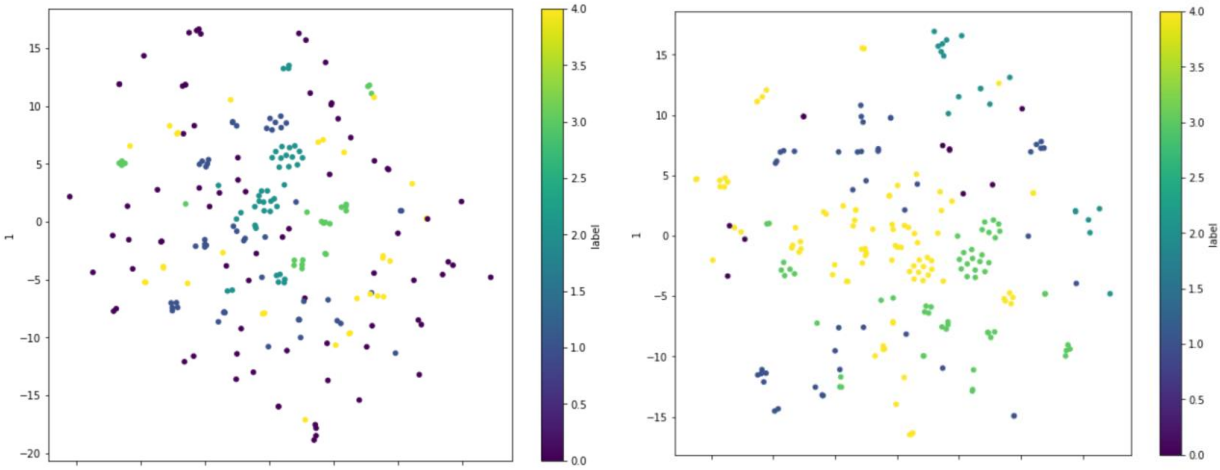


Figure 18 the clustering outputs in scatter plot of K-means with DTW on the left and kernel-K-means on the right, except the different colouring, these two figures are very alike

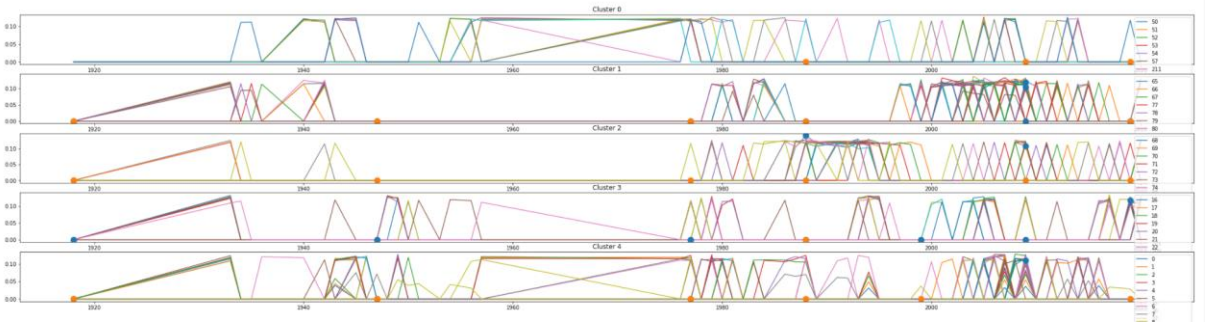


Figure 19 time series clusters and binders based on kernel-K-means, as K-means demonstrated in figure 12, similar time series are clustered together but binders disperse. Weak binders mutate over time but the strong ones don't mutate

Since there isn't a significant difference in clustering outputs, we decided to employ K-means with DTW for its theoretical advantage in clustering time series based on their similar shapes.

Conclusion

In conclusion, the purpose of this project is to verify if the host immune system leads to distinctive mutation patterns in the evolutionary history of viruses. Firstly, we decided to conduct our research with the influenza A virus, especially the H1N1 subtype, for its long history of evolution for one hundred years. The host immune system interferes with the viral invasion in two different branches: the innate and the adaptative branches. The innate immune response works in a more rapid and non-specific way. On the contrary, the adaptative one takes effect in several days, eliminates specific invaded pathogens and memorizes them through T and B lymphocytes. One of the T lymphocytes members killer T cells can precisely recognize infected cells and destroy them. In addition, killer T cells recognize epitopes presented on the surface of infected cells. But the epitopes that antibodies generated by B lymphocytes bind with is untraceable due to antibodies' 3D structure as all proteins. Therefore, our study should base on H1N1 viruses' epitopes binding to killer T cells. In particular, we chose NS1 protein for its promising performance in binding with T cell receptors according to previous research, and the allele HLA-A02:01 since it is the most common HLA allele.

The main challenge of this research is to devise ways of encoding information about how individual residues and groups of neighbouring residues mutate over time. To accomplish this mission, after loaded and pre-processed the protein sequences, we have tested the majority of existing sequence embedding libraries. Eventually, we have chosen the Sequence Graph Transform library (SGT) to embed the 9-mers for its capacity of coding short- and long-term patterns while keeping a low level of computation. Only embedding the 9-mer segments is not enough to code information about mutations of single and groups of residues over time. In this research, we are interested in understanding the rhythm of evolution, which means ignoring the specific year and amino acid type of the mutation, but coding its frequency and positions. Inspired by one "pandas" method that generates the difference between a DataFrame element and the previous one, we decided to calculate the Euclidean distance between a 9-mer embedding and its previous one since the embeddings with SGT are in Euclidean space. If this distance equals 0, it means no mutation occurred in the current year. Using this method, we have successfully interpreted the frequency of mutations, and its position is the distance presented above. Thus, we have obtained a set of time series about the evolution of 9-mers in history.

Previously we have transformed the clustering problem into a time series clustering problem. For this purpose, we employed the "tslearn" library that provides time series clustering and classification tools. However, the clustering result doesn't appear to be satisfying: indeed, 9-mers sharing the same mutational patterns gathered together, but the predicted variable epitopes disperse in almost every cluster. To prove that the pressures from host immune systems have led to distinctive mutational patterns, such epitopes are expected to appear in only one or two clusters since the 9-mers of the same pattern gather together. Hence, we have no choice but to admit that with the current methods of sequence embedding and transformation into time series we cannot prove such pressure from the host immune system has led to distinctive mutation patterns. The reasons for such results are still hard to determine. For now, we have several hypotheses: it could be because the clustering cannot generate a strong separation between predicted variable epitopes and other peptides, or some non-epitope

peptides present the same properties as the real epitopes. In addition, we still doubt that there could be more sequence embedding methods. For instance, there is a sequence embedding library which is only released recently called “bio-embeddings”. As one of the consequences, this library presents many imperfections like dependencies conflicts. Moreover, there is little feedback which means it is rarely employed. But it is possible that with its improvements in the future, it would be able to solve our problem.

Nevertheless, we don’t consider this result a failure since the reasons remain uncertain, and the duration of this internship has restricted any further progress. We conclude that firstly, we have correctly selected and processed the data from a massive database. Secondly, despite the possibility of choosing a less efficient data encoding method by generating time series from sequence embeddings, it is still the best method that we can devise for the moment. Finally, as a personal achievement, I had the opportunity to put my skills and knowledge gained at school into practice by working individually for three months. I have tested numerous methods to find a solution. Without a doubt, this internship helped me to gain the ability to search for relevant information and to solve machine learning problems independently. In addition, I appreciate the opportunity to explore the biology area because it has been my interest since high school. Thanks to this internship, I have gained much interesting biologic knowledge. More importantly, I have been able to solve problems using my machine learning skills.

Glossary

Immune response: a series of reactions that occurs in an organism to defend itself from the invasion from pathogens

Pathogen: an organism that parasitizes a host and causes diseases

APC: Antigen-presenting cell, a type of immune cell, in this research we focus on its ability to present antigens to T-lymphocyte cells

T-lymphocyte cells: a type of immune cells, together with other immune cells, T cells kill infected cells and memorize antigens

B-lymphocyte cells: a type of immune cells, one of its kinds – plasma cells generate antibodies to bind with antigens

MHC: major histocompatibility complex, a large linkage group of DNA containing polymorphic genes. These genes code for proteins located on almost all nucleated cells to present pathogens peptides to killer T cells. They are called MHC molecules, and they play a crucial role in the adaptive immune system

Substitution: a type of DNA replication error that places the wrong nucleotide or sequence of nucleotides in the wrong position, which results in non-synonymous mutations that change the amino acid type

9-mer: the length of the MHC class I groove

Epitope: a partial peptide of an antigen that killer T cells can recognize during immune response

Allele: a variant of a gene, one or multiple alleles control the different forms of the same trait

FASTA: a text-based format to represent nucleotide and protein sequences. Both nucleotides and amino acids types are represented by single-letter codes

NLP: Natural language processing, an artificial intelligence subfield for computer to study human language

Time series: a sequence of chronologically sorted data

References

[1] ‘Tracking SARS-CoV-2 variants’. https://www.who.int/emergencies/emergency-health-kits/trauma-emergency-surgery-kit-who-tesk-2019/tracking-SARS-CoV-2-variants (accessed Aug. 28, 2021).

[2] J. S. Tregoning, K. E. Flight, S. L. Higham, Z. Wang, and B. F. Pierce, ‘Progress of the COVID-19 vaccine effort: viruses, vaccines and variants versus efficacy, effectiveness and escape’, *Nat. Rev. Immunol.*, pp. 1–11, Aug. 2021, doi: 10.1038/s41577-021-00592-1.

[3] ‘Birkbeck, University of London’, *Wikipedia*. Aug. 28, 2021. Accessed: Sep. 03, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Birkbeck,_University_of_London&oldid=1041086576

[4] ‘Achievements and strengths — Birkbeck, University of London’. https://www.bbk.ac.uk/departments/biology/about-us/ (accessed Sep. 03, 2021).

[5] ‘Immune response’, *Wikipedia*. Aug. 21, 2021. Accessed: Aug. 30, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Immune_response&oldid=1039882797

[6] ‘Adaptive immune system’, *Wikipedia*. Aug. 13, 2021. Accessed: Aug. 31, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Adaptive_immune_system&oldid=1038665823

[7] ‘Humoral immunity’, *Wikipedia*. Aug. 23, 2021. Accessed: Aug. 31, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Humoral_immunity&oldid=1040318968

[8] ‘Major histocompatibility complex’, *Wikipedia*. Aug. 02, 2021. Accessed: Aug. 31, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Major_histocompatibility_complex&oldid=1036812864

[9] ‘Epitope’, *Wikipedia*. Aug. 28, 2021. Accessed: Aug. 31, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Epitope&oldid=1041056801

[10] M. Denyer, ‘The prediction of immunodominant T-cell epitopes and analysis of their conservation’, 2014.

[11] ‘Substitution Mutation: Definition, Examples, Types’, *Biology Dictionary*, Jun. 20, 2018. https://biologydictionary.net/substitution-mutation/ (accessed Sep. 08, 2021).

[12] Y. Zhang *et al.*, ‘Influenza Research Database: An integrated bioinformatics resource for influenza virus research’, *Nucleic Acids Res.*, vol. 45, no. Database issue, pp. D466–D474, Jan.

2017, doi: 10.1093/nar/gkw857.

[13] '*Influenza A virus*', *Wikipedia*. Sep. 03, 2021. Accessed: Sep. 03, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Influenza_A_virus&oldid=1042123921

[14] 'FASTA format', *Wikipedia*. Sep. 03, 2021. Accessed: Sep. 04, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=FASTA_format&oldid=1042093956

[15] 'pandas (software)', *Wikipedia*. Jul. 02, 2021. Accessed: Sep. 04, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Pandas_(software)&oldid=1031574126

[16] M. Nielsen and M. Andreatta, 'NetMHCpan-3.0; improved prediction of binding to MHC class I molecules integrating information from multiple receptor and peptide length datasets', *Genome Med.*, vol. 8, no. 1, p. 33, Dec. 2016, doi: 10.1186/s13073-016-0288-x.

[17] C. Ranjan, 'Sequence Embedding for Clustering and Classification', *Medium*, May 04, 2019. https://towardsdatascience.com/sequence-embedding-for-clustering-and-classification-f816a66373fb (accessed Sep. 04, 2021).

[18] 'Embeddings | Machine Learning Crash Course', *Google Developers*. https://developers.google.com/machine-learning/crash-course/embeddings/video-lecture?hl=fr (accessed Sep. 04, 2021).

[19] A. Hayes, 'Understanding Time Series', *Investopedia*. https://www.investopedia.com/terms/t/timeseries.asp (accessed Sep. 04, 2021).

[20] 'Time Series Clustering — tslearn 0.5.2 documentation'. https://tslearn.readthedocs.io/en/stable/user_guide/clustering.html#k-means-and-dynamic-time-warping (accessed Sep. 13, 2021).

[21] D. M. J. Garbade, 'Understanding K-means Clustering in Machine Learning', *Medium*, Sep. 12, 2018. https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1 (accessed Sep. 05, 2021).

[22] 'pandas.DataFrame.diff — pandas 1.3.2 documentation'. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.diff.html (accessed Sep. 12, 2021).

[23] L. Derksen, 'Visualising high-dimensional datasets using PCA and t-SNE in Python', *Medium*, Apr. 29, 2019. https://towardsdatascience.com/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b (accessed Sep. 11, 2021).

[24] L. Impressions 3d, 'Applications médicales de l'impression 3D', Jan. 26, 2019. https://www.lesimpressions3d.com/medical-applications-of-3d-printing/ (accessed Sep. 14, 2021).