**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**──────── \* ───────**



**BÁO CÁO**

**Project cuối kỳ**

***Đề tài:***

**Sử dụng đặc tính spike protein để dự đoán nguy cơ lây nhiễm và theo dõi động thái tiến hóa của coronavirus**

**Giảng viên hướng dẫn: TS. Nguyễn Hồng Quang**

**Sinh viên thực hiện: (Your name)**

**Lớp: (Your class)**

**Hà Nội, 08/2021**

# Nhóm sinh viên thực hiện

# Giáo viên hướng dẫn

TS. Nguyễn Hồng Quang,

Giảng viên Bộ môn Kỹ thuật máy tính,

Trưởng phòng thí nghiệm Tin học y sinh, Trung tâm nghiên cứu Quốc tế về Trí tuệ Nhân tạo (BK.AI),

Viện Công nghệ thông tin và Truyền thông,

Trường Đại học Bách Khoa Hà Nội

**Tóm tắt:** Coronavirus có thể vượt qua hàng rào loài và lây nhiễm cho con người gây ra triệu chứng về đường hô hấp một cách nghiêm trọng. SARS-CoV-2 khả năng lớn bắt nguồn từ loài dơi và vẫn đang gây bệnh cho người ở Trung Quốc. Trong nghiên cứu này, một mô hình dự doán được đề xuất đề đánh giá nguy cơ lây nhiễm của coronavirus không bắt nguồn từ người để có cảnh báo sớm.

**Từ khóa:** Deep learning, ProtBert, Random Forest, classification, Transfer leaning.

# 1. Giới thiệu (Introduction)

Virus Corona rất phổ biến ở cả người và động vật với hội chứng hô hấp cấp nghiêm trọng. SARS-CoV-2 được cho là có nguồn gốc là loài dơi ở Vũ Hán, Trung Quốc. Virus có vỏ bọc (enveloped virus) này gồm một sợi RNA dương bản (+). Hầu hết các virus hoàn chỉnh có hình cầu với các protein gai (spike glucoprotein, còn gọi là S protein) gắn trên vỏ ngoài. Ngoài ra, virus còn có các protein cấu trúc khác bao gồm protein Envelope (E), Matrix (M) và Nucleocapsid (N).

Họ Coranaviridae bao gồm 4 chi (genera), alpha-, beta-, delta- và gammacoronavirus, cũng như một vài phân chi (subgenera) và loài (species). Những phân tích về phát sinh chủng loại học (phylogenetic analysis) trên bộ gen của virus Corona cho thấy SARS-CoV-2 là thành viên với của chi betacoronavirus. Chi này cũng bao gồm virus Corona gây nên Hội chứng hô hấp cấp tính nặng (SARS-CoV), virus gây Hội chứng hô hấp cấp Trung Đông (Middle East Respiratory Syncdrome-related coronavirus – MERS-CoV), Hội chứng hô hấp cấp tính nặng từ dơi (bat SARS-related coronaviruses –SARSr-CoV), cũng như những loài virus khác đã được tìm thấy ở người và ở nhiều loài động vật đa dạng khác. Việc lây truyền virus Corona trong cùng loài động vật và giữa các loài khác nhau cùng với sự tái tổ hợp di truyền góp phần vào sự xuất hiện của các chủng virus Corona mới.

Qua đây, chúng ta thấy rằng khả năng đột biến của các chủng virus Corona là rất lớn. Trong bài báo Origin and evolution of pathogenic coronaviruses [3] (Nguồn gốc và sự tiến hóa của những virus corona gây bệnh) đã chỉ ra rằng SARS-CoV-2 và MERS-CoV có khả năng bắt nguồn từ dơi, cả hai loài này lây nhiễm cho người thông qua các vật chủ trung gian khác nhau. Bài báo Identifying SARS-CoV-2 related coronaviruses in Malayan pangolins [4] (Xác định các virus corona liên quan đến SARS-CoV-2 ở tê tê Malaysia) xuất bản năm 2020 cũng đặt ra giả thuyết về việc liệu tê tê Malaysia có phải là một vật chủ trung gian của virus Corona. Trong bài báo đề cập đến việc các nhà khoa học đã phát hiện các virus Corona lây nhiễm trên tê tê, trong đó bao gồm một loài virus có sự tương đồng mạnh mẽ với SARS-CoV-2 trong miền liên kết thụ thể (receptor-binding domain – RBD) trên protein S. Các bài báo trên đều sử dụng các phương pháp truyền thống để phát hiện virus corona như các xét nghiệm PCR (qPCR) Vì thế trong báo cáo này đã đưa ra một mô hình dự đoán dựa trên học máy (Machine Learning). Để đánh giá nguy cơ lây nhiễm của coronavirus không có nguồn gốc của con người để cảnh báo sớm.

Với bộ dữ liệu gồm 2666 trình tự protein virus corona được phân tích trong nghiên cứu có sẵn trong Cơ sở dữ liệu 2019nCovR của NGDC, bọn em sử dụng phương pháp rừng ngẫu nhiên trong học máy để có thể dự đoán sự lây nhiễm chéo loài của virus corona. Mã nguồn được viết bởi ngôn ngữ lập trình python. Và sử dụng bộ ba thuật toán G-gap dipeptide composition (GGAP), Pseudo-amino-acid composition (PseAAC), Amino acid composition (AAC) mã hóa các thuộc tính quan trọng của protein đột biến, đó là thông tin thành phần, thông tin liên quan đến vị trí và các đặc tính hóa lý.

Các kết quả đã đạt được:

Kết quả của 10-fold cross-validation (phương pháp kiểm tra độ chính xác của một máy học dựa trên tập dữ liệu cho trước) cho thấy hiệu suất đã đạt được rất tốt khi sử dụng thuật toán GGAP với g = 13. Với mô hình dự đoán này đã đạt được AAC tối đa là 98,74% cùng với hệ số tương quan Matthews (MCC) là 97,28%.

# 2. Mô tả bài toán

## 2.1. Chi tiết bài toán.

Bài toán đặt ra:

* Đầu vào (Input):

Dữ liệu đầu vào là 2666 trình tự protein coronavirus được phân tích trong nghiên cứu có sẵn trong cơ sở dữ liệu 2019nCoVR của NGDC. Dữ liệu được lưu trữ trong tệp Data.xlsx đính kèm. Dữ liệu gốc gồm 6 trường:

Phenotype (kiểu hình) với giá trị H là có nguồn gốc từ người, N là không có nguồn gốc từ người.

ID (mã định danh): mã của trình tự protein trên NCBI.

Nomenclature là danh mục tên gọi của trình tự protein.

Location: Nơi trình tự được phát hiện.

Host: Nguồn gốc loài của trình tự.

Date: ngày giải mã trình tự.

Dữ liệu đầu vào gốc qua quá trình từ mã định danh của protein được tổng hợp thành đầu vào của mô hình học máy dưới dạng fasta.

Tệp fasta đầu vào có định dạng như sau:

><id của trình tự protein>@<phenotype của protein>

<trình tự chi tiết của protein>

Ví dụ:

>ACJ35486@H

MFLILLISLPMALAVIGDLKCTTVSINDVDTGVPSISTDTVDVTNGLGTYYVLDRVYLNTTLLLNGYYPTSGSTYRNMALKGTLLLSTLWFKPPFLSDFTNGIFAKVKNTKVTKNGLMYSEFPAITIGSTFVNTSYSVVVQPHTTNLDNKLQGLLEISVCQYTMCEYPNTICHPNLGNRRVELWHWDTGVVSCLYKRNFTYDVNADYLYFHFYQEGGTFYAYFTDTGVVTKFLFNVYLGTVLSHYYVMPLTCNSAMSLEYWVTPLTSKQYLLAFNQDGVIFNAVDCKSDFMSEIKCKTLSIAPSTGVYELNGYTVQPIADVYRRIPNLPDCNIEAWLNDKSVPSPLNWERKTFSNCNFNMSSLMSFIQADSFTCNNIDAAKIYGMCFSSITIDKFAIPNSRKVDLQLGNLGYLQSFNYRIDTTATSCQLYYNLPAANVSVSRFNPSTWNRRFGFTEQSVFKPQPAGVFTDHDVVYAQHCFKAPTNFCPCKLDGSLCVGNGPGIDAGYKNSGIGTCPAGTNYLTCHNAAQCNCLCTPDPITSKSTGPYKCPQTKYLVGIGEHCSGLAIKSDHCGGNPCTCQPQAFLGWSVESCLQGDRCNIFANFILHGVNSGTTCSTDLQKSNTDIILGVCVNYDLYGITGQGIFVEVNATYYNSWQNLLYDSNGNLYGFRDYLTNRTFMIRSCYSGRVSAAFHANSSEPALLFRNIKCNYVFNNTLSRQLQPINYFDSYLGCVVNADNSTSSVVQTCDLTVGSGYCVDYSTKRRSRRAITTGYRFTNFEPFTVNSVNDSLEPVGGLYEIQIPSEFTIGNMEEFIQTSSPKVTIDCSAFVCGDYAACKSQLVEYGSFCDNINAILTEVNELLDTTQLQVANSLMNGVTLSTKLKDGVNFNVDDINFSPVLGCLGSDCNRVSSRSAIEDLLFSKVKLADVGFVEAYNNCTGGAEIRDLICVQSYNGIKVLPPLLSENQISGYTLAATSASLFPPWSAAAGVPFYLNVQYRINGIGVTMDVLSQNQKLIANAFNNALGAIQEGFDATNSALVKIQAVVNANAEALNNLLQQLSNRFGAISSSLQEILSRLDALEAQAQIDRLINGRLTALNAYVSQQLSDSTLVKFSAAQAMEKVNECVKSQSSRINFCGNGNHIISLVQNAPYGLYFIHFSYVPTKYVTAKVSPGLCIAGDRGIAPKSGYFVNVNNTWMFTGSGYYYPEPITGNNVVVMSTCAVNYTKAPDVMLNISTPNLPDFKEELDQWFKNQTSVAPDLSLDYINVTFLDLQEEMNRLQEAIKVLNHSYINLKDIGTYEYYVKWPWYVWLLIGFAGVAMLVLLFFICCCTGCGTSCSKKCGGCCDDYTGHQELVIKTSHDD.

* Đầu ra (Output): kết quả dự báo khả năng lây nhiễm virus corona sang người: giá trị 1 ứng với khả năng lây nhiễm; giá trị 0 ứng với khả năng không lây nhiễm.

## 2.2. Tập dữ liệu sử dụng.

Dữ liệu được lấy từ cơ sở dữ liệu 2019nCoVR của NGDC [5]. NGDC là Trung tâm Dữ liệu Genomics Quốc gia của Trung Quốc, là một phần của Trung tâm Thông tin tin Sinh học Quốc gia Trung Quốc. Sau khi lấy được dữ liệu từ cơ sở dữ liệu và mã định danh của chúng (ID), danh sách các mã định danh trình tự protein sẽ được đưa vào một chương trình crawl dữ liệu từ NCBI để tải về trình tự với định dạng fasta của chúng. NCBI là viết tắt của cụm từ National Center for Biotechnology (Trung tâm công nghệ Thông tin Công nghệ sinh học Quốc gia Hoa Kỳ), là một đơn vị trực thuộc Thư viện Y học Quốc gia Hoa Kỳ, trực thuộc Viện Y tế Quốc gia Hoa Kỳ. NCBI là nơi lưu trữ nguồn tài nguyên liên quan tới công nghệ sinh học và y học, các cơ sở dữ liệu chính của NCBI bao gồm GenBank (thông tin về các trình tự ADN), PubMed (cơ sở dữ liệu trích dẫn các tài liệu về sinh và y học). Sau đây là những dữ liệu được sử dụng (các tệp đính kèm báo cáo).

* Danh sách coronavirus được mô tả trong tập dữ liệu Data\_raw.xlsx.
* Dữ liệu về trình tự protein của 2666 virus corona Data.fasta.

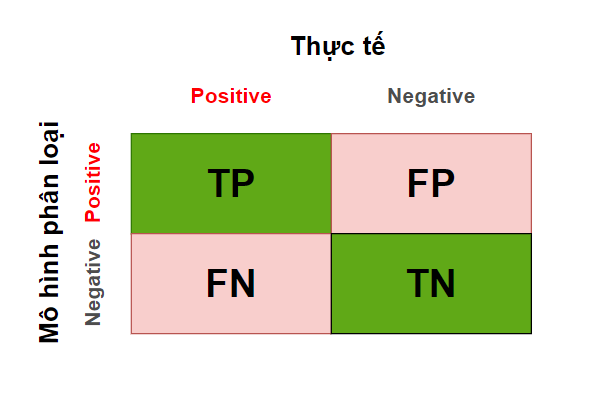
Các chủng được đưa ra trong dữ liệu có bộ gen dài và đầy đủ, được phân lập từ năm 1941 đến năm 2020, trong đó bao gồm cả các chủng SARS-CoV-2.

Trong bộ 2666 trình tự protein này có bao gồm tất cả 507 mẫu là có nguồn gốc từ con người được coi là positive (dương tính) còn 2159 mẫu còn lại không có nguồn gốc từ con người được coi là nagative (âm tính).

Với bộ dữ liệu khá nhỏ chỉ khoảng 2666 nên tôi dùng phương pháp k-fold cross validation để ước lượng hiệu quả của mô hình học máy, cụ thể là 10-fold cross validation. Tập dữ liệu sau khi xáo trộn ngẫu nhiên được chia thành 10 phần có kích thước gần bằng nhau và không có phần tử chung. Giá trị k = 10 là một cấu hình k-fold rất phổ biến và được chứng minh là cho sai số nhỏ, phương sai thấp. Tại mỗi lần chạy kiểm tra thì một trong 10 phần sẽ được chọn ra làm tập validation, 9 tập dữ liệu còn lại là tập train được dùng để huấn luyện mô hình.

## 2.3. Các độ đo đánh giá

Tôi sử dụng 4 thang đo chính để đánh giá độ hiệu quả và chính xác của mô hình, đó là: TPR (true positive rate hay còn gọi là recall, sensitivity hoặc độ nhạy), TNR (true negative rate hay còn gọi là specificity hoặc độ đặc hiệu); ACC (accuracy, độ chính xác); MCC (Hệ số tương quan Matthews).



Hình 1: Confusision matric

### True Positive rate (TPR)

Tỷ lệ phân loại positive đúng trên tổng số các trường hợp positive. Tỷ lệ này còn có tên gọi khác là sensitivity (độ nhạy), hit rate (tỷ lệ trúng đích) hay recal. Tỷ lệ này dược tính như sau:

### True negative rate (TNR)

Tỷ lệ loại trừ đúng trên tổng số các trường hợp negative hay còn gọi là specification (độ đặc hiệu). Tỷ lệ này dược tính như sau:

Ý nghĩa của Positive/Negative là tương đối tương tự cho Sensitivity/Specificity. Chúng đều đo lường độ hữu dụng của mô hình, hay khả năng cho phép đáp ứng mục tiêu phát hiện/loại trừ.

### Accuracy (ACC)

Tỷ lệ của tất cả trường hợp phân loại đúng (không phân biệt negative/positive) trên toàn bộ trường hợp trong kiểm định. Tỷ lệ này dược tính như sau:

Đây là tiêu chí phổ biến nhất (thường được nghĩ đến đầu tiên) khi kiểm định hiệu năng của mô hình phân loại, tuy nhiên giá trị thực dụng của nó thường kém vì nó không đặc hiệu cho một mục tiêu nào cả.

### Hệ số tương quan Matthews (MCC)

MCC là một phương pháp riêng của giới Machine learning, dùng để đánh giá phẩm chất của mô hình phân loại nhị phân. Mục tiêu của MCC là khắc phục vấn đề dữ liệu bị mất cân bằng. MCC có bản chất là một hệ số tương quan giữa giá trị Thực tế và Kết quả dự báo của mô hình, nó được xác định như sau:

Giá trị của MCC dao động trong khoảng -1 đến +1, MCC = +1 biểu thị cho “kết quả phân loại hoàn hảo”, MCC = 0 cho thấy mô hình vô dụng (không hơn gì sự phán đoán ngẫu nhiên) còn MCC = -1 cho thấy mô hình không những vô dụng hoàn toàn mà còn tuyệt đối sai, vì kết quả phân loại hoàn toàn trái nghịch với quan sát thực tế.

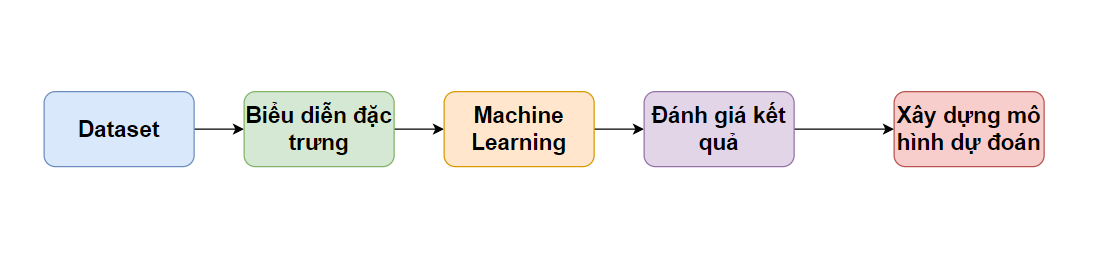
# 3. Các hướng nghiên cứu liên quan (Related Works)

Bài báo “Evolutionary study of COVID‐19, severe acute respiratory syndrome coronavirus 2 (SARS‐CoV‐2) as an emerging coronavirus: Phylogenetic analysis and literature review” đã nghiên cứu về sự tiến hóa của Covid-19 cũng như phân tích phát sinh loài và tổng quan tài liệu về loại virus này. Mục tiêu của bài báo trên là từ những bài báo đã được các tác giả tổng hợp về trình tự protein của covid-19 hoặc những trình tự protein của virus có họ hàng gần với covid-19 để có thể tạo lên cây phát sinh loài của của covid-19 từ đó có thể dự đoán nguồn gốc, nghiên cứu tiến hóa của covid-19. Phương pháp được bài báo sử dụng để xây dựng nên cây phát sinh loài là Neighbor Joining sử dụng 1000 bootstrap. Trong tin sinh học, neighbor joining là một phương pháp phân cụm bottom-up để tạo ra cây phát sinh loài, được tạo ra bởi Naruya Saitou và Masatoshi Nei, nó thường được sử dụng cho cây dựa trên dữ liệu trình tự DNA hoặc protein. Bài báo sử dụng trình tự protein của các virus để tính khoảng các giữa các cặp đơn vị phân loại để hình thành cây. Nghiên cứu trên đã cung cấp một loạt các chi, dòng và chủng corona virus khác nhau để có thể hiểu rõ hơn về dịch tễ học phân tử của corona virus. Từ đó đưa ra nhận xét SARS-CoV-2 là một tác nhân gây bệnh từ động vật và tổ tiên gần của nó có thể là coronavirus liên quan đến SARS. Corona virus được phân loại là Sarbecovirus nhánh con của betacoronavirus.

Trong bài “The evolutionary history of ACE2 usage within the coronavirus subgenus Sarbecovirus” chỉ ra rằng mô hình sử dụng ACE2 được quan sát trong số các sarbecovirus được giải thích tốt nhất bằng sự tái tổ hợp không phải của SARS-CoV-2 mà là của SARS-CoV-1 và các họ hàng của nó. Bài báo đề cập đến sự xuất hiện gần đây của coronavirus gây hội chứng hô hấp cấp tính nghiêm trọng 2 (SARS-CoV-2) ở Trung Quốc và sự lây lan nhanh chóng của nó trên khắp thế giới chứng tỏ coronavirus (CoV) từ động vật hoang dã vẫn là mối đe dọa cấp bách đối với sức khỏe cộng đồng toàn cầu và sự ổn định kinh tế. Đặc biệt, coronavirus từ chi Sarbecovirus (bao gồm SARS-CoV-2, SARS-CoV-1, nhiều loại virus dơi và một số lượng nhỏ virus tê tê) được coi là nhóm có nguy cơ cao cho sức khỏe con người. Vì cả hai sarbecovirus đã gây bệnh ở người (SARS-CoV-1 và -2) đều sử dụng men chuyển 2 (ACE2) làm thụ thể tế bào của chúng, sự tiến hóa của thụ thể này có tầm quan trọng đặc biệt trong việc tìm hiểu con đường xuất hiện của sarbecovirus. Coronavirus giống dơi SARS Rp3 là họ hàng gần về mặt di truyền học của SARS-CoV-1 nhưng không thể liên kết với ACE2 (hACE2) ở người trong ống nghiệm. Ngược lại, những họ hàng gần khác của SARS-CoV-1, bao gồm coronavirus giống dơi SARS WIV1 và WIV16, có khả năng liên kết với hACE2. Một số loại virus giống SARS-CoV-1 khác cũng đã được kiểm tra khả năng sử dụng hACE2 và so sánh trình tự protein đột biến của chúng cho thấy rằng các vi rút không thể sử dụng hACE2 một cách đồng bộ có một hoặc hai lần xóa trong vùng liên kết thụ thể của chúng (RBD) khiến chúng có cấu trúc rất khác so với những vi rút sử dụng hACE2. Mục đích chính của bảo báo là cho thấy rằng SARS-CoV-1 và các họ hàng gần của nó đã đạt được khả năng sử dụng ACE2 thông qua sự kiện tái tổ hợp xảy ra giữa tổ tiên của SARS-CoV-1 và vi rút dòng 5 về mặt phát sinh chủng loại. SARS-CoV-2, chỉ có thể xảy ra với các dòng họ chiếm cùng một không gian địa lý và vật chủ.

Đến với bài báo “Evolutionary study of COVID‐19, severe acute respiratory syndrome coronavirus 2 (SARS‐CoV‐2) as an emerging coronavirus: Phylogenetic analysis and literature review” đánh giá sự phát sinh và tiến hóa của vi rút SARS ‐ CoV ‐ 2 bằng cách sử dụng vi rút hiện tại và các trình tự liên quan. Hầu hết các đặc điểm virus học và dữ liệu lâm sàng liên quan đến SARS ‐ CoV ‐ 2 đã được thiết lập bởi công trình có giá trị của Chan, Yuan, et al. (Năm 2020). Họ đã thực hiện một nghiên cứu được công bố vào tháng 1 năm 2020, trong đó họ báo cáo tổ chức bộ gen và phân loại SARS ‐ CoV ‐ 2 nhờ sự hỗ trợ của Công nghệ giải trình tự Oxford Nanopore. Kết quả của họ cho thấy virus này có liên quan đến virus SARS giống như coronavirus của dơi. Virus này cũng được xếp vào dòng B của Betacoronavirus. Chiều dài bộ gen ước tính khoảng 29,8 kb có hàm lượng GC là 38%. Phân tích axit amin của SARS ‐ CoV ‐ 2 cho thấy tiểu đơn vị 1 của protein S và vùng liên kết thụ thể lõi có độ tương đồng lần lượt là 66% và 68% với coronavirus liên quan đến SARS. Cấu trúc bộ gen của SARS ‐ CoV ‐ 2 gần như tương tự với SARS của dơi và các vi rút liên quan đến SARS khác với sự khác biệt nhỏ. Bộ gen bao gồm sáu ORF phổ biến (1a / b, H, S, E, M và N) và sáu ORF phụ (bao gồm 3a, 6, 7a, 7b, 8 và 10. Sự sắp xếp trình tự axit amin cho các protein không cấu trúc của SARS ‐ CoV ‐ 2 và dơi SARS ‐ như coronavirus (phân lập dơi ‐ SL ‐ CoVZXC21) cho thấy 88% –99% nhận dạng đối với các protein khác nhau và giữa SARS ‐ CoV ‐ 2 và SARS nó là 68% –100%. Việc xác định các biến thể và chủng mới trong coronavirus động vật, đặc biệt là dơi, rất quan trọng để hiểu được sự tiến hóa của coronavirus.

# 4. Đề xuất mô hình



Hình 2: Mô hình xử lý

Mô hình bao gồm 2 phần chính là biểu diễn tính năng và sử dụng học máy. Cụ thể, biểu diễn thuộc tính được sử dụng với 3 thuật toán là “Amino acid composition – AAC”, “Parallel correlation-based pseudo-amino-acid composition - PC-PseAAC”; “G-gap dipeptide composition – GGAP”. Sau đó, các thuộc tính sẽ trở thành đầu vào cho mô hình học máy với phương pháp rừng ngẫu nhiên. Phương pháp rừng ngẫu nhiên được sử dụng để huấn luyện và kiểm tra trên tập dữ liệu và đưa ra dự đoán về nguy cơ lây nhiễm của virus corona.

## 4.1. Tập dữ liệu

Sử dụng 2666 chuỗi protein gai của virus corona được thu thập như trên phần 2. Trong đó, gồm 507 virus có nguồn gốc từ người và được được coi là mẫu dương tính, còn lại 2159 virus không bắt nguồn từ ngược và được coi là mẫu âm tính.

## 4.2. Biểu diễn thuộc tính

Để biểu diễn các thông tin về gai protein, tôi sử dụng 3 thuật toán mã hóa protein với nhiều khía cạnh khác nhau như thông tin về thành phần, thông tin về vị trí tương đối và thông tin về tính chất hóa lý. Các thuật toán mã hóa được mô tả chi tiết như sau:

### Composition axit amin – Thành phần axit amin - ACC

* Thành phần axit amin là đặc điểm đơn giản nhưng thường được sử dụng để phân tích trình tự và xây dựng mô hình. ACC mô tả tần suất của từng loại axit amin trong chuỗi protein được tính bằng:
* Trong đó:
* là số lần xuất hiện axit amin cùng loại
* là chiều dài của chuỗi
* Với thuật toán này, tôi tạo vector có chiều dài 20 ứng với tần suất của 20 loại axit amin khác nhau với mỗi chuỗi protein.

### Parallel correlation-based pseudo-amino-acid composition – PseACC

* Thuật toán ACC sử dụng phương pháp thống kê để trích xuất lại đặc điểm của chuỗi protein do vậy thông tin về tứ tự chuỗi sẽ bị mất đi. Do đó, thuật toán PseACC đã giải quyết được vấn đề này, nó kết hợp được thông tin về thống kê, thông tin về vị trí và đặc điểm lý hóa của amino axit trong chuỗi.
* Thuật toán PseAAC sẽ tạo ra vector có độ dài 20 + λ:

Với

* Trong đó:
* là số nguyên
* ứng với giá trị chuẩn hóa tần suất xuất hiện của 20 loại amino axit xuất hiện trong protein P
* biểu diễn bậc tương quan cao nhất trong trình tự protein P.
* là giá trị hàm tương quan của tương quan bậc j với tất cả giá trị tương quan trong P. được tính theo biểu thức sau:
* ứng với độ phân cực, cấu trúc bậc hai, khối lượng phân tử, sự đa dạng codon và điện tích tương ứng với axit amin thứ P trong trình tự protein P.
* Thuật toán PseAAC tạo ra tối đa 20 vector thuộc tính có kích thước và giá trị khác nhau ứng với giá trị đối với mỗi chuỗi protein.

### G-gap dipeptide composition - GGAP

* Thuật toán GGAP mô tả mối tương quan giữa 2 axit amin xuất hiện trong chuỗi protein.
* Thuật toán PseAAC sẽ tạo ra vector có độ dài 400:
* Trong đó
* là tần suất xuất hiện của u-th gap dipeptide trong chuỗi protein và được tính:
* là số lần xuất hiện u-th gap dipeptide trong chuỗi protein
* Thuật toán GGAP tạo ra tối đa 20 vector thuộc tính có kích thước và giá trị khác nhau ứng với giá trị đối với mỗi chuỗi protein.

Dưới đây là bảng tổng kết về vector biểu diễn thuộc tính sử dụng 3 thuật toán đã trình bày trên.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Loại | Chiều | STT | Loại | Chiều |
| 1 | PseAAC = 1 | 21 | 21 | GGAP g = 0 | 400 |
| 2 | PseAAC = 2 | 22 | 22 | GGAP g = 1 | 400 |
| 3 | PseAAC = 3 | 23 | 23 | GGAP g = 2 | 400 |
| 4 | PseAAC = 4 | 24 | 24 | GGAP g = 3 | 400 |
| 5 | PseAAC = 5 | 25 | 25 | GGAP g = 4 | 400 |
| 6 | PseAAC = 6 | 26 | 26 | GGAP g = 5 | 400 |
| 7 | PseAAC = 7 | 27 | 27 | GGAP g = 6 | 400 |
| 8 | PseAAC = 8 | 28 | 28 | GGAP g = 7 | 400 |
| 9 | PseAAC = 9 | 29 | 29 | GGAP g = 8 | 400 |
| 10 | PseAAC = 10 | 30 | 30 | GGAP g = 9 | 400 |
| 11 | PseAAC = 11 | 31 | 31 | GGAP g = 10 | 400 |
| 12 | PseAAC = 12 | 32 | 32 | GGAP g = 11 | 400 |
| 13 | PseAAC = 13 | 33 | 33 | GGAP g = 12 | 400 |
| 14 | PseAAC = 14 | 34 | 34 | GGAP g = 13 | 400 |
| 15 | PseAAC = 15 | 35 | 35 | GGAP g = 14 | 400 |
| 16 | PseAAC = 16 | 36 | 36 | GGAP g = 15 | 400 |
| 17 | PseAAC = 17 | 37 | 37 | GGAP g = 16 | 400 |
| 18 | PseAAC = 18 | 38 | 38 | GGAP g = 17 | 400 |
| 19 | PseAAC = 19 | 39 | 39 | GGAP g = 18 | 400 |
| 20 | PseAAC = 20 | 40 | 40 | GGAP g = 19 | 400 |
|  |  |  | 41 | AAC | 20 |
|  |  |  |  |  |  |

## 4.3. Học máy

Sau khi áp dụng thuật toán để biểu diễn thuộc tính của gai protein, ta thu được 41 cách biểu diễn thuộc tính cho 2666 chuỗi gai protein ban đầu. Từ 41 cách biểu diễn này, tôi sử dụng thuật toán rừng ngẫu nhiên để xây dựng mô hình và đưa ra dự đoán cho nguy cơ lây nhiễm bênh từ virus corona không bắt nguồn sang người.

Thuật toán rừng ngẫu nhiên là thuật toán học có giám sát. Thuật toán này có thể được sử dụng cho cả phân lớp và hồi quy. Nó cũng là thuật toán linh hoạt và dễ sử dụng nhất. Thuật toán rừng ngẫu nhiên tạo ra cây quyết định trên các mẫu dữ liệu được chọn ngẫu nhiên, được dự đoán từ mỗi cây và chọn giải pháp tốt nhất bằng cách bỏ phiếu. Nó còn có thể cung cấp một chỉ báo khá tốt về tầm quan trọng của thuộc tính.

Giả sử bộ dữ liệu của mình có n dữ liệu (sample) và mỗi dữ liệu có d thuộc tính. Tóm tắt thuật toán rừng ngẫu nhiên với các bước như sau:

Bước 1: Lấy ngẫu nhiên x dữ liệu từ bộ dữ liệu với kĩ thuật Bootstrapping, hay còn gọi là random sampling with replacement.

Bước 2: Sau khi sample được x dữ liệu từ bước 1 thì mình chọn ngẫu nhiên ở y thuộc tính (y < d). Giờ mình được bộ dữ liệu mới gồm x dữ liệu và mỗi dữ liệu có y thuộc tính.

Bước 3: Dùng thuật toán cây quyết định để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

Bước 4: Lặp lại các bước trên để xây dựng đủ số cây. Sau đó kết quả dự đoán sẽ được tổng hợp từ các cây quyết định.

Cụ thể, khi áp dụng cho bài toán, tôi xây dựng 500 cây đối với mỗi thuộc tính đang được sử dụng.

## 4.4. Huấn luyện và kiểm thử.



Hình 3: Phương pháp rừng ngẫu nhiên

Tôi sử dung phương pháp K-fold cross validation để ước lượng độ hiệu quả của mô hình học máy và chọn ra mô hình tốt nhất cho bài toán.

Phương pháp này thường bao gồm các bước như sau:

Xáo trộn dataset một cách ngẫu nhiên

Chia dataset thành k nhóm

Với mỗi nhóm:

Sử dụng nhóm hiện tại để đánh giá hiệu quả mô hình

Các nhóm còn lại được sử dụng để huấn luyện mô hình

Huấn luyện mô hình

Đánh giá và sau đó hủy mô hình

Tổng hợp hiệu quả của mô hình dựa từ các số liệu đánh giá

Trong quá trình huấn luyện các fold, tôi sẽ lưu lại mô hình tốt nhất để làm mô hình dự đoán cho bài toán.

Cụ thể, tôi sử dụng với 10-fold. Tuy nhiên, do tập dữ liệu không cân bằng về số lượng mẫu dương và số lượng mẫu âm nên tôi đã trực tiếp nhân đôi số lượng mẫu dương trước khi chia.

# 5. Thực hiện hệ thống

## 5.1. Xử lý dữ liệu đầu vào

Dữ liệu đầu vào được định dạng FASTA có thể chứa nhiều chuỗi protein. Mỗi chuỗi gồm 2 phần là mô tả và dữ liệu của trình tự.

Đầu tiên, dòng mô tả được phân biệt với dữ liệu bằng ký tự lớn hơn (“>”) ở đầu, có dạng: “>SeqID@{H|N}”. Trong đó:

SeqID là định danh của chuỗi protein trong cơ sở dữ liệu NCBI (National Center for Biotechnology Information);

Chữ cái H hoặc N có ý nghĩa là mẫu dương hay mẫu âm;

Chữ “@” là ngăn cách giữa hai mô tả trên.

Sau đó xuống dòng là dữ liệu chứa các ký tự mô tả trình tự axit amin trong chuỗi protein có độ dài phụ thuộc vào protein đã được định danh bởi SeqID.

Ví dụ với một chuỗi protein sau đây:

>ACJ35486@H

MFLILLISLPMALAVIGDLKCTTVSINDVDTGVPSISTDTVDVTNGLGTYYVLDRVYLNTTLLLNGYYPTSGSTYRNMALKGTLLLSTLWFKPPFLSDFTNGIFAKVKNTKVTKNGLMYSEFPAITIGSTFVNTSYSVVVQPHTTNLDNKLQGLLEISVCQYTMCEYPNTICHPNLGNRRVELWHWDTGVVSCLYKRNFTYDVNADYLYFHFYQEGGTFYAYFTDTGVVTKFLFNVYLGTVLSHYYVMPLTCNSAMSLEYWVTPLTSKQYLLAFNQDGVIFNAVDCKSDFMSEIKCKTLSIAPSTGVYELNGYTVQPIADVYRRIPNLPDCNIEAWLNDKSVPSPLNWERKTFSNCNFNMSSLMSFIQADSFTCNNIDAAKIYGMCFSSITIDKFAIPNSRKVDLQLGNLGYLQSFNYRIDTTATSCQLYYNLPAANVSVSRFNPSTWNRRFGFTEQSVFKPQPAGVFTDHDVVYAQHCFKAPTNFCPCKLDGSLCVGNGPGIDAGYKNSGIGTCPAGTNYLTCHNAAQCNCLCTPDPITSKSTGPYKCPQTKYLVGIGEHCSGLAIKSDHCGGNPCTCQPQAFLGWSVESCLQGDRCNIFANFILHGVNSGTTCSTDLQKSNTDIILGVCVNYDLYGITGQGIFVEVNATYYNSWQNLLYDSNGNLYGFRDYLTNRTFMIRSCYSGRVSAAFHANSSEPALLFRNIKCNYVFNNTLSRQLQPINYFDSYLGCVVNADNSTSSVVQTCDLTVGSGYCVDYSTKRRSRRAITTGYRFTNFEPFTVNSVNDSLEPVGGLYEIQIPSEFTIGNMEEFIQTSSPKVTIDCSAFVCGDYAACKSQLVEYGSFCDNINAILTEVNELLDTTQLQVANSLMNGVTLSTKLKDGVNFNVDDINFSPVLGCLGSDCNRVSSRSAIEDLLFSKVKLADVGFVEAYNNCTGGAEIRDLICVQSYNGIKVLPPLLSENQISGYTLAATSASLFPPWSAAAGVPFYLNVQYRINGIGVTMDVLSQNQKLIANAFNNALGAIQEGFDATNSALVKIQAVVNANAEALNNLLQQLSNRFGAISSSLQEILSRLDALEAQAQIDRLINGRLTALNAYVSQQLSDSTLVKFSAAQAMEKVNECVKSQSSRINFCGNGNHIISLVQNAPYGLYFIHFSYVPTKYVTAKVSPGLCIAGDRGIAPKSGYFVNVNNTWMFTGSGYYYPEPITGNNVVVMSTCAVNYTKAPDVMLNISTPNLPDFKEELDQWFKNQTSVAPDLSLDYINVTFLDLQEEMNRLQEAIKVLNHSYINLKDIGTYEYYVKWPWYVWLLIGFAGVAMLVLLFFICCCTGCGTSCSKKCGGCCDDYTGHQELVIKTSHDD

ACJ35486 là SeqID;

H thể hiện protein này là mẫu dương;

Từ dòng thứ 2 là trình tự axit amin.

Với dữ liệu đầu vào ở trên, tôi thực hiện 3 thuật toán mã hóa thuộc tính đã mô tả chi tiết trong phần 4.2 để thu được vector thuộc tính tương ứng với chuỗi protein đầu vào.

### Composition axit amin – Thành phần axit amin – ACC

Sau đây là hàm thực hiện mã hóa theo thuật toán ACC.

Đầu vào là các chuỗi protein và mảng được quan tâm với chỉ số “order” để có thể thay đổi thứ tự axit amin của thuật toán mã hóa.

Đầu ra của hàm là các vector đặc trưng cho chuỗi protein và header mô tả thứ tự ứng với đầu vào.

def AAC(fastas, \*\*kw):

    AA = kw['order'] if kw['order'] != None else 'ACDEFGHIKLMNPQRSTVWY'

    encodings = []

    header = ['#']

    for aa in AA:

        header.append(aa)

    encodings.append(header)

    for i in fastas:

        name, sequence = i[0], i[1]

        code = [name]

        length = len(sequence)

        for aa in AA:

            fre = sequence.count(aa) / length

            code.append(fre)

        encodings.append(code)

    return encodings, header

Sau khi thực hiện đọc, sử dụng thuật toán mã hóa và ghi file, ta được một file ghi lại vector đặc trưng của chuỗi như sau với dòng đầu tiên mô tả thứ tự mã hóa; dòng thứ 2 với chỉ số đầu tiên là số thứ tự của protein trong có trong file FASTA và 20 số tiếp theo là giá trị của vector đặc trưng. Dưới đây là vector đặc trưng ứng với protein có SeqID là ACJ35486 đã mô tả ở phần 5.1.

,A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y

0,0.05942773294203962,0.041085840058694055,0.049889948642699924,0.030814380044020543,0.05062362435803375,0.06823184152604549,0.012472487160674981,0.055025678650036686,0.03961848862802641,0.09317681584739546,0.015407190022010272,0.08217168011738811,0.04035216434336023,0.03815113719735877,0.02641232575201761,0.08290535583272193,0.07410124724871607,0.0748349229640499,0.011005135730007337,0.05429200293470286

### Parallel correlation-based pseudo-amino-acid composition – PseACC

Sau đây là hàm thực hiện mã hóa theo thuật toán PseACC.

Đầu vào là các chuỗi protein; lambdaValue và w tương ứng với giá trị lambda và weight; mảng được quan tâm với chỉ số “order” để có thể thay đổi thứ tự axit amin của thuật toán mã hóa.

Đầu ra của hàm là các vector đặc trưng cho chuỗi protein và header mô tả thứ tự ứng với đầu vào.

def Rvalue(aa1, aa2, AADict, Matrix):

    return sum([(Matrix[i][AADict[aa1]] - Matrix[i][AADict[aa2]]) \*\* 2 for i in range(len(Matrix))]) / len(Matrix)

def PseAAC(fastas, lambdaValue, w=0.05, \*\*kw):

    # Read data

    with open("PAAC.txt") as f:

        records = f.readlines()

    AA = ''.join(records[0].rstrip().split()[1:])

    AADict = {}

    for i in range(len(AA)):

        AADict[AA[i]] = i

    AAProperty = []

    AAPropertyNames = []

    for i in range(1, len(records)):

        array = records[i].rstrip().split(

        ) if records[i].rstrip() != '' else None

        AAProperty.append([float(j) for j in array[1:]])

        AAPropertyNames.append(array[0])

    AAProperty1 = []

    for i in AAProperty:

        meanI = sum(i) / 20

        fenmu = math.sqrt(sum([(j-meanI)\*\*2 for j in i])/20)

        AAProperty1.append([(j-meanI)/fenmu for j in i])

    #

    encodings = []

    header = ['#']

    for aa in AA:

        header.append('Xc1.' + aa)

    for n in range(1, lambdaValue + 1):

        header.append('Xc2.lambda' + str(n))

    encodings.append(header)

    encodings = []

    header = ['#']

    for aa in AA:

        header.append('Xc1.' + aa)

    for n in range(1, lambdaValue + 1):

        header.append('Xc2.lambda' + str(n))

    encodings.append(header)

    for i in fastas:

        name, sequence = i[0], re.sub('-', '', i[1])

        code = [name]

        theta = []

        for n in range(1, lambdaValue + 1):

            theta.append(

                sum([Rvalue(sequence[j], sequence[j + n], AADict, AAProperty1) for j in range(len(sequence) - n)]) / (

                    len(sequence) - n))

            myDict = {}

        for aa in AA:

            myDict[aa] = sequence.count(aa)

        code = code + [myDict[aa] / (1 + w \* sum(theta)) for aa in AA]

        code = code + [(w \* j) / (1 + w \* sum(theta)) for j in theta]

        encodings.append(code)

    return encodings, header

Sau khi thực hiện chương trình, ta thu được vector đặc trưng tương ứng. Sau đây là ví dụ về vector đặc trưng bằng phương pháp PseAAC với giá trị lambda = 1:

,Xc1.A,Xc1.R,Xc1.N,Xc1.D,Xc1.C,Xc1.Q,Xc1.E,Xc1.G,Xc1.H,Xc1.I,Xc1.L,Xc1.K,Xc1.M,Xc1.F,Xc1.P,Xc1.S,Xc1.T,Xc1.W,Xc1.Y,Xc1.V,Xc2.lambda1

0,74.79649239470527,33.242885508757894,103.42231047169123,62.79211707209825,51.71115523584562,48.01750129042807,38.783366426884214,85.8774542309579,15.698029268024563,69.25601147657895,117.27351276700702,49.86432826313685,19.391683213442107,63.715530558452635,50.78774174949123,104.34572395804562,93.26476212179298,13.85120229531579,68.33259799022457,94.18817560814738,0.076586513645614

### G-gap dipeptide composition - GGAP

Sau đây là hàm thực hiện mã hóa theo thuật toán GGAP.

Đầu vào là các chuỗi protein; “gap” là đối số thể hiện khoảng cách giữa axit amin trong chuỗi; và mảng “kw” được quan tâm với chỉ số “order” để có thể thay đổi thứ tự axit amin của thuật toán mã hóa.

Đầu ra của hàm là các vector đặc trưng cho chuỗi protein và header mô tả thứ tự ứng với đầu vào.

def GGAP(fastas, gap, \*\* kw):

    AA = kw['order'] if kw['order'] != None else 'ACDEFGHIKLMNPQRSTVWY'

    encodings = []

    header = ['#']

    patterns = []

    for aa1 in AA:

        for aa2 in AA:

            header.append(aa1 + aa2)

            patterns.append(aa1 + "[" + AA + "]" + "{" + str(gap) + "}" + aa2)

    encodings.append(header)

    for i in fastas:

        name, sequence = i[0], i[1]

        code = [name]

        length = len(sequence)

        denominator = length - gap - 1

        for pattern in patterns:

            fre = len(re.findall(pattern, sequence)) / denominator

            code.append(fre)

        encodings.append(code)

    return encodings, header

Sau khi thực hiện, ta thu được các vector đặc trưng cho chuỗi protein với vector có chiều dài 400.

## 5.2. Thực hiện hệ thống

Như đã mô tả ở phần 5.1, sau khi biểu diễn thuộc tính của dữ liệu, ta thu được 41 cách biểu diễn. Tôi thực hiện phương pháp rừng ngẫu nhiên trên 41 cách biểu diễn này và so sánh hiệu quả đạt được của từng cách biểu diễn.

Hàm sau thực hiện đọc dữ liệu và trực tiếp nhân đôi mẫu dương.

def prepare\_data(factor):

    X = pd.read\_csv(

        "../FeatureExtraction/GGAP/GGAP\_data\_3.csv", index\_col=0)

    y = pd.read\_csv("./target\_data.csv", index\_col=0)

    reps = [factor if val == 1 else 1 for val in y.target]

    X = X.loc[np.repeat(X.index.values, reps)]

    y = y.loc[np.repeat(y.index.values, reps)]

    return np.array(X), np.array(y)

Sau khi chuẩn bị dữ liệu, ta huấn luyện và kiểm tra mô hình bằng phương pháp K-fold validation.

    X, y = prepare\_data(2)

    kf = KFold(n\_splits=10, shuffle=True)

    print(X.shape, y.shape)

    kf.get\_n\_splits(X)

    sn = []

    sp = []

    ACC = []

    MCC = []

    for train\_index, test\_index in kf.split(X):

        X\_train, X\_test = X[train\_index], X[test\_index]

        y\_train, y\_test = y[train\_index], y[test\_index]

        # Set the random state for reproducibility

        fit\_rf = RandomForestClassifier(n\_estimators=500)

        fit\_rf.fit(X\_train, y\_train.ravel())

        y\_pred = fit\_rf.predict(X\_test)

        print("\n===============")

        tn, fp, fn, tp = confusion\_matrix(y\_test, y\_pred).ravel()

        print("SN:",  tp /(tp + fn))

        sn.append(tp /(tp + fn))

        print("SP:", tn / (tn + fp))

        sp.append(tn / (tn + fp))

        print("ACC:", accuracy\_score(y\_test, y\_pred))

        ACC.append(accuracy\_score(y\_test, y\_pred))

        print("MCC:", matthews\_corrcoef(y\_test, y\_pred))

        MCC.append(matthews\_corrcoef(y\_test, y\_pred))

    print("\n Summary")

    print("SN: min = {:.4f} & max =  {:.4f}".format(np.min(sn), np.max(sn)))

    print("SP: min = {:.4f} & max =  {:.4f}".format(np.min(sp), np.max(sp)))

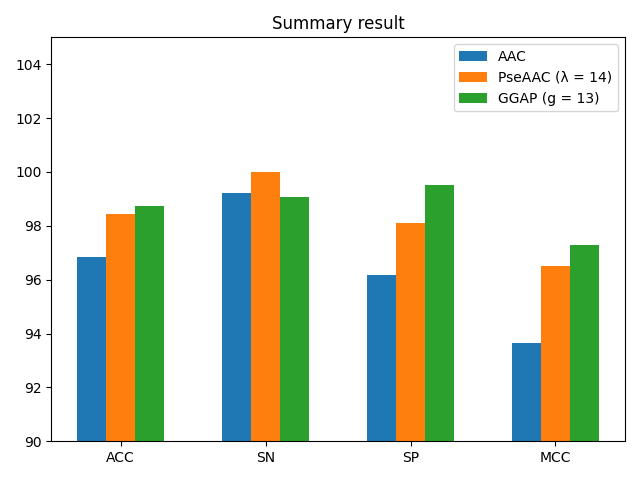
    print("ACC: min = {:.4f} & max =  {:.4f}".format(np.min(ACC), np.max(ACC)))

    print("MCC: min = {:.4f} & max =  {:.4f}".format(np.min(MCC), np.max(MCC)))

## 5.3. Xử lý kết quả đầu ra

Đầu ra của mô hình là giá trị 1 hay 0 tương ứng cho kết quả dự đoán khả năng lây nhiễm virus corona sang người.

Dưới đây là biểu đồ thể hiện kết quả tốt nhất cho mỗi thuật toán mã hóa bằng phương pháp 10-fold validation.



Hình 4: Kết quả dự báo tốt nhất trên các phương pháp trích xuất đặc trưng

# 6. Thử nghiệm và kết quả

## 6.1. Môi trường thử nghiệm

Chương trình được thử nghiệm trên môi trường Google Colab. Đây là một môi trường notebook Jupyter được host, sử dụng miễn phí và không cần thiết lập trên Google. Với cấu hình như sau:

* CPU: Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz
* RAM: 8 GB

Chương trình sử dụng các thư viện sau:

* Matplotlib
* Numpy
* Pandas
* Sklearn
* Transformers
* Tensorflow
* Và một số thư viện: re, sys, os, …

## 6.2. Các thử nghiệm

## a. Thử nghiệm trích xuất đặc trưng bằng amino axit composition, Parallel correlation-based pseudo-amino-acid composition và G-gap dipeptide composition

Như đã mô tả phương pháp trích xuất đặc trưng ở phần 5.1, tôi thu được các vector đặc trưng tương ứng. Từ đây tôi chuẩn bị dữ liệu cho quá trình huấn huyện bằng hàm sau:

def prepare\_data(factor, feature\_type, value\_type):

    if (feature\_type == "AAC"):

        X = pd.read\_csv(

            "../FeatureExtraction/AAC/AAC\_data.csv", index\_col=0)

        print("../FeatureExtraction/AAC/AAC\_data.csv")

    elif (feature\_type == "PseAAC"):

        X = pd.read\_csv(

            "../FeatureExtraction/PseAAC/PseAAC\_data\_lambda\_{}.csv".format(value\_type), index\_col=0)

        print("../FeatureExtraction/PseAAC/PseAAC\_data\_lambda\_{}.csv".format(value\_type))

    elif (feature\_type == "GGAP"):

        X = pd.read\_csv(

            "../FeatureExtraction/GGAP/GGAP\_data\_gap\_{}.csv".format(value\_type), index\_col=0)

        print("../FeatureExtraction/GGAP/GGAP\_data\_gap\_{}.csv".format(value\_type))

    else:

        assert False, "Feature path is not exist"

    y = pd.read\_csv("./target\_data.csv", index\_col=0)

    reps = [factor if val == 1 else 1 for val in y.target]

    X = X.loc[np.repeat(X.index.values, reps)]

    y = y.loc[np.repeat(y.index.values, reps)]

    return np.array(X), np.array(y)

Hàm nhận 4 tham số:

factor là cơ số nhân mẫu dương.

feature\_type là kiểu đặc trưng

value\_type là giá trị thay đổi tương ứng với kiểu đặc trưng.

Hàm trên sẽ trả về đặc trưng tương ứng và nhãn.

Sau khi chuẩn bị dữ liêu, tôi xây dựng hàm cho quá trình huấn luyện ứng với 41 thuộc tính đã trích xuất. Sau khi huấn luyện, tôi ghi kết quả ra file để thuận tiện cho việc đánh giá.

    over\_sn = []

    over\_sp = []

    over\_ACC = []

    over\_MCC = []

    ret\_sn, ret\_sp, ret\_acc, ret\_mcc = train\_test("AAC", None)

    over\_sn.append(ret\_sn)

    over\_sp.append(ret\_sp)

    over\_ACC.append(ret\_acc)

    over\_MCC.append(ret\_mcc)

    for i in range(1, 21):

        ret\_sn, ret\_sp, ret\_acc, ret\_mcc = train\_test("PseAAC", i)

        over\_sn.append(ret\_sn)

        over\_sp.append(ret\_sp)

        over\_ACC.append(ret\_acc)

        over\_MCC.append(ret\_mcc)

    for i in range(0, 20):

        ret\_sn, ret\_sp, ret\_acc, ret\_mcc = train\_test("GGAP", i)

        over\_sn.append(ret\_sn)

        over\_sp.append(ret\_sp)

        over\_ACC.append(ret\_acc)

        over\_MCC.append(ret\_mcc)

    with open("summary.txt", "a") as f:

        f.write("\n Overall")

        f.write("SN: min = {:.4f} & max =  {:.4f}".format(np.min(over\_sn), np.max(over\_sn)))

        f.write("SP: min = {:.4f} & max =  {:.4f}".format(np.min(over\_sp), np.max(over\_sp)))

        f.write("ACC: min = {:.4f} & max =  {:.4f}".format(np.min(over\_ACC), np.max(over\_ACC)))

        f.write("MCC: min = {:.4f} & max =  {:.4f}".format(np.min(over\_MCC), np.max(over\_MCC)))

Như vậy, sau quá trình này tôi thu được kết quả sau:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Loại | SN | SP | ACC | MCC | Loại | SN | SP | AAC | MCC |
| PseAAC = 1 | 1.0000 | 0.9630 | 0.9748 | 0.9446 | GGAP g = 0 | 1.0000 | 0.9765 | 0.9779 | 0.9510 |
| PseAAC = 2 | 1.0000 | 0.9677 | 0.9780 | 0.9520 | GGAP g = 1 | 1.0000 | 0.9762 | 0.9811 | 0.9584 |
| PseAAC = 3 | 1.0000 | 0.9680 | 0.9748 | 0.9437 | GGAP g = 2 | 0.9901 | 0.9860 | 0.9811 | 0.9573 |
| PseAAC = 4 | 1.0000 | 0.9762 | 0.9748 | 0.9439 | GGAP g = 3 | 1.0000 | 0.9778 | 0.9779 | 0.9472 |
| PseAAC = 5 | 1.0000 | 0.9648 | 0.9717 | 0.9343 | GGAP g = 4 | 1.0000 | 0.9906 | 0.9811 | 0.9582 |
| PseAAC = 6 | 0.9898 | 0.9621 | 0.9654 | 0.9242 | GGAP g = 5 | 1.0000 | 0.9818 | 0.9842 | 0.9634 |
| PseAAC = 7 | 1.0000 | 0.9704 | 0.9779 | 0.9531 | GGAP g = 6 | 0.9909 | 0.9727 | 0.9780 | 0.9524 |
| PseAAC = 8 | 1.0000 | 0.9631 | 0.9748 | 0.9460 | GGAP g = 7 | 0.9905 | 0.9696 | 0.9717 | 0.9312 |
| PseAAC = 9 | 1.0000 | 0.9689 | 0.9748 | 0.9409 | GGAP g = 8 | 0.9907 | 0.9712 | 0.9748 | 0.9449 |
| PseAAC = 10 | 1.0000 | 0.9731 | 0.9717 | 0.9333 | GGAP g = 9 | 0.9902 | 0.9680 | 0.9717 | 0.9408 |
| PseAAC = 11 | 0.9906 | 0.9818 | 0.9811 | 0.9558 | GGAP g = 10 | 1.0000 | 0.9764 | 0.9842 | 0.9654 |
| PseAAC = 12 | 1.0000 | 0.9683 | 0.9717 | 0.9384 | GGAP g = 11 | 1.0000 | 0.9685 | 0.9748 | 0.9422 |
| PseAAC = 13 | 0.9906 | 0.9717 | 0.9748 | 0.9440 | GGAP g = 12 | 1.0000 | 0.9709 | 0.9686 | 0.9307 |
| PseAAC = 14 | 1.0000 | 0.9810 | 0.9843 | 0.9652 | GGAP g = 13 | 0.9907 | 0.9950 | 0.9874 | 0.9728 |
| PseAAC = 15 | 1.0000 | 0.9633 | 0.9716 | 0.9378 | GGAP g = 14 | 0.9909 | 0.9763 | 0.9779 | 0.9510 |
| PseAAC = 16 | 1.0000 | 0.9665 | 0.9717 | 0.9387 | GGAP g = 15 | 1.0000 | 0.9733 | 0.9811 | 0.9559 |
| PseAAC = 17 | 0.9896 | 0.9769 | 0.9717 | 0.9352 | GGAP g = 16 | 1.0000 | 0.9752 | 0.9842 | 0.9668 |
| PseAAC = 18 | 0.9904 | 0.9801 | 0.9811 | 0.9594 | GGAP g = 17 | 1.0000 | 0.9771 | 0.9748 | 0.9467 |
| PseAAC = 19 | 0.9903 | 0.9767 | 0.9779 | 0.9500 | GGAP g = 18 | 0.9904 | 0.9738 | 0.9780 | 0.9512 |
| PseAAC = 20 | 0.9905 | 0.9630 | 0.9716 | 0.9370 | GGAP g = 19 | 0.9905 | 0.9729 | 0.9716 | 0.9381 |
| AAC | 0.9921 | 0.9617 | 0.9686 | 0.9364 |  |  |  |  |  |

# 7. Thảo luận

Như vậy, sau khi thử nghiệm nhiều phương pháp đã mô tả ở phần 6, tôi đã xây dựng được mô hình dự đoán khả năng lây nhiễm virus corona sang người gây ra bệnh viêm phổi cấp bằng phương pháp rừng ngẫu nhiên. Kết quả được đưa ra với phương pháp này đạt đến AAC tối đa là 98,74% cùng với hệ số tương quan Matthews (MCC) là 97,28%. Với kết quả này trên tập dữ liệu đã xử lý và các vấn đề có liên quan, tôi đảm bảo kết quả này có tính cạnh tranh với các xử lý trong quá khứ hoặc trong tương lai sau này.

Đóng góp chính của tôi vào giải quyết bài toán là:

Đề xuất sử dụng gai protein để dự đoán khả năng lây nhiễm.

So sánh các phương pháp trích xuất đặc trưng và chọn ra phương pháp trích xuất đặc trưng hiệu quả.

Đề ra mô hình dự đoán khả năng lây nhiễm virus corona.

Quy trình từ chuỗi gai protein đến lúc dự đoán ra kết quả.

Do đây là bài đầu tiên giải quyết vấn đề đã đặt ra nên tôi hi vọng từ các đóng góp này sẽ giúp phần nào giải quyết bài toán dự đoán khả năng lây nhiễm virus corona sang người và định hướng cho các nghiên cứu liên quan về sau.

Với kết quả ACC lên đến 98.74%, nên hầu hết các chuỗi gai protein được đưa vào dự đoán có kết quả đúng với mong đợi. Kết quả hầu như chấp nhận được trong quá trình dự đoán vì có một số yếu tố mà trong quá trình trích xuất đặc trưng bằng phương pháp GGAP với g = 13 không thể bao quát được tất cả các trường hợp. Tôi đã thử nghiệm với các phương pháp trích xuât đặc trưng phức tạp hơn như ProtBert, tuy nhiên kết quả với thử nghiệm này chưa đạt được hiệu quả thực sự cao. Có thể do quá trình chọn chiều dài chuỗi để đưa vào mô hình ProtBert và quá trình turning các tham số của mạng phân loại với đầu vào là vector có chiều dài 1024 ứng với mỗi chuỗi protein đầu vào. Tôi đã thử nhiều phương pháp phân loại nhưng kết quả chỉ đạt được độ chính xác ACC tối đa là 86% với phương pháp rừng ngẫu nhiên là 82%.

# 8. Tổng kết và phương hướng phát triển (Conclusions and Perspectives)

Tôi đã mô tả quá trình trích xuất đặc trưng từ chuỗi gai protein. Cho đến khi sử dụng các đặc trưng vào mô hình học máy, cụ thể phương pháp rừng ngẫu nhiên với kết quả thu được tốt nhất. Phương pháp trích xuất đặc trưng tuy đơn giản kết hợp với phương pháp rừng ngẫu nhiên đã thể hiện kết quả dự đoán vượt trội trên tập dữ liệu kiểm tra.

Tuy rằng kết quả chưa được tốt nhưng tôi mong rằng thử nghiệm này sẽ giúp mọi người có thêm một lựa chọn trích xuất đặc trưng của chuỗi để áp dụng vào bài toán của mình. Cụ thể, đối với bài toán “Dự đoán nguy cơ lây nhiễm virus corona sang người” chúng ta có thể đưa qua một số mô hình phân loại khác phù hợp với dữ liệu thu thập được và đúng với yêu cầu sẽ thu được kết quả khả quan hơn.

# 9. Tài liệu tham khảo

[1] Using the spike protein feature to predict infection risk and monitor the evolutionary dynamic of coronavirus; 2020 Mar 25;

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7093988/>

[2] ProtTrans: Towards Cracking the Language of Life’s Code Through Self-Supervised Deep Learning and High Performance Computing;

<https://doi.org/10.1101/2020.07.12.199554>

[3] Evolutionary study of COVID‐19, severe acute respiratory syndrome coronavirus 2 (SARS‐CoV‐2) as an emerging coronavirus: Phylogenetic analysis and literature review; 2021 Mar;

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7753621/>

[4] Classical Coronaviruses; 2020 April 30;

<https://link.springer.com/chapter/10.1007%2F978-981-15-4814-7_12>

[5] The evolutionary history of ACE2 usage within the coronavirus subgenus Sarbecovirus; 2021 Jan;

[https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7928622/](%20https:/www.ncbi.nlm.nih.gov/pmc/articles/PMC7928622/)

[6] Evolutionary study of COVID‐19, severe acute respiratory syndrome coronavirus 2 (SARS‐CoV‐2) as an emerging coronavirus: Phylogenetic analysis and literature review; 2020 Nov 18;

[https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7753621/#vms3394-bib-0022](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7753621/%23vms3394-bib-0022)

[7] Composition and divergence of coronavirus spike proteins and host ACE2 receptors predict potential intermediate hosts of SARS‐CoV‐2; 2020 Mar 11;

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7228221/>

[8] Structural basis of receptor recognition by SARS-CoV-2; 2020 Mar 30;

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7328981/>

[9] Properties of Coronavirus and SARS-CoV-2; 2020 Apr;

<https://pubmed.ncbi.nlm.nih.gov/32342926/>

[10] MLP-Mixer: An all-MLP Architecture for Vision; 2021 May 4;

<https://arxiv.org/abs/2105.01601>

[11] Comparison of Severe Acute Respiratory Syndrome Coronavirus 2 Spike Protein Binding to ACE2 Receptors from Human, Pets, Farm Animals, and Putative Intermediate Hosts; 2020 Jul 16;

<https://pubmed.ncbi.nlm.nih.gov/32404529/>

[12] A Novel Coronavirus from Patients with Pneumonia in China, 2019; 2020 Feb 20;

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7092803/>

[13] Prediction of protein cellular attributes using pseudo-amino acid composition;

<https://pubmed.ncbi.nlm.nih.gov/11288174/>

[14] COVID-19 pandemic: Insights into structure, function, and hACE2 receptor recognition by SARS-CoV-2

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7444525/>

[15] Amino acid.

<https://en.wikipedia.org/wiki/Amino_acid>

[16] Zhao WM, Song SH, Chen ML, Zou D, Ma LN, Ma YK, et al. The 2019 novel coronavirus resource

<https://pubmed.ncbi.nlm.nih.gov/32102777/>

[17] Clinical Characteristics of Coronavirus Disease 2019 in China

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7266766/>

[18] The Significant and Profound Impacts of Chou’s Pseudo Amino Acid Composition or PseAAC

<https://www.scirp.org/journal/paperinformation.aspx?paperid=102834>

# Mục lục

[Nhóm sinh viên thực hiện 2](#_Toc80573792)

[Giáo viên hướng dẫn 2](#_Toc80573793)

[1. Giới thiệu (Introduction) 3](#_Toc80573794)

[2. Mô tả bài toán 4](#_Toc80573795)

[2.1. Chi tiết bài toán. 4](#_Toc80573796)

[2.2. Tập dữ liệu sử dụng. 6](#_Toc80573797)

[2.3. Các độ đo đánh giá 7](#_Toc80573798)

[a. True Positive rate (TPR) 8](#_Toc80573799)

[b. True negative rate (TNR) 8](#_Toc80573800)

[c. Accuracy (ACC) 8](#_Toc80573801)

[d. Hệ số tương quan Matthews (MCC) 8](#_Toc80573802)

[3. Các hướng nghiên cứu liên quan (Related Works) 9](#_Toc80573803)

[4. Đề xuất mô hình 12](#_Toc80573804)

[4.1. Tập dữ liệu 12](#_Toc80573805)

[4.2. Biểu diễn thuộc tính 12](#_Toc80573806)

[a. Composition axit amin – Thành phần axit amin - ACC 13](#_Toc80573807)

[b. Parallel correlation-based pseudo-amino-acid composition – PseACC 13](#_Toc80573808)

[c. G-gap dipeptide composition - GGAP 14](#_Toc80573809)

[4.3. Học máy 16](#_Toc80573810)

[4.4. Huấn luyện và kiểm thử. 17](#_Toc80573811)

[5. Thực hiện hệ thống 18](#_Toc80573812)

[5.1. Xử lý dữ liệu đầu vào 18](#_Toc80573813)

[a. Composition axit amin – Thành phần axit amin – ACC 19](#_Toc80573814)

[b. Parallel correlation-based pseudo-amino-acid composition – PseACC 20](#_Toc80573815)

[c. G-gap dipeptide composition - GGAP 22](#_Toc80573816)

[5.2. Thực hiện hệ thống 23](#_Toc80573817)

[5.3. Xử lý kết quả đầu ra 24](#_Toc80573818)

[6. Thử nghiệm và kết quả 25](#_Toc80573819)

[6.1. Môi trường thử nghiệm 25](#_Toc80573820)

[6.2. Các thử nghiệm 25](#_Toc80573821)

[a. Thử nghiệm trích xuất đặc trưng bằng amino axit composition, Parallel correlation-based pseudo-amino-acid composition và G-gap dipeptide composition 25](#_Toc80573822)

[7. Thảo luận 28](#_Toc80573823)

[8. Tổng kết và phương hướng phát triển (Conclusions and Perspectives) 30](#_Toc80573824)

[9. Tài liệu tham khảo 30](#_Toc80573825)

[Mục lục 33](#_Toc80573826)

[Lưu ý: 34](#_Toc80573827)

# Lưu ý:

* Không sử dụng lại các hình vẽ của các nghiên cứu khác. Cần vẽ các hình vẽ theo cách hiểu của riêng mình.
* Không copy lại quá 6 từ liên tiếp từ các báo cáo khác (kể cả chuyển ngữ từ tiếng Anh sang tiếng Việt).
* Các hình vẽ và bảng biểu cần có chú giải chi tiết (tối thiểu 2 dòng) và cần có tham chiếu đến hình vẽ trong đoạn văn tương ứng.
* Các hình vẽ ngoài việc đưa vào báo cáo thì cần để ở định dạng pdf ở các file riêng và nộp kèm theo báo cáo.
* Khuyến khích vẽ các sơ đồ ở trang <https://app.diagrams.net/> và nộp file mã nguồn vẽ sơ đồ kèm theo báo cáo.