به نام خدا

فاز ۳ پروژه‌ی الگوریتم‌های بیوانفورماتیک

نحوه‌ی اجرای کد:

پکیج‌های مورد نیاز برای اجرای کد، در فایل requirements.txt آمده است.

ساخت درخت فیلوژنتیک:

برای ساخت درخت فیلوژنتیک باید از فایل phlogeny.py استفاده کنید. با استفاده از این فایل میتوانید Distance Matrix بین ژنوم‌ها را با استفاده از score الگوریتم CHROMEISTER تولید کنید. همچنین می‌توان با استفاده از این ماتریس، درخت فیلوژنتیک را با دو روش UPGMA و Neighbor Joining خروجی گرفت. همچنین می‌توان ماتریس را به فرمت meg. برای استفاده در نرم‌افزار Mega خروجی گرفت.

آرگومان‌های ورودی به شرح زیر می‌باشد (وارد کردن آرگومان‌های قرمز اجباری می‌باشد):

- h-: نمایش پیغام راهنمایی در مورد آرگومان‌های ورودی
- mode: حالت compute برای محاسبه‌ی ماتریس فاصله از روی ژنوم‌ها است. پس از محاسبه خروجی‌ها نیز تولید می‌شوند. در حالت result تنها با استفاده از ماتریس فاصله ورودی، خروجی‌ها تولید می‌شوند.
- --meta: آدرس فایل json مربوط به ژنوم‌هایی که در تولید ماتریس فاصله استفاده شده‌اند. (فایل‌های استفاده شده برای نمونه فرستاده شده است.)
- --dist-mat: آدرس فایل pickle مربوط به ماتریس فاصله. در حالت compute ماتریس بدست آمده در این آدرس ذخیره می‌شود و در حالت result از این آدرس خوانده می‌شود.
- --genomes-dir: آدرس فولدر محل ژنوم‌ها. تنها در حالت compute کاربرد دارد.
- --plot: آدرس فایل ذخیره سازی نمودار درخت فیلوژنتیک.
- --plot-title: عنوان نمودار.

- no-show--: عدم نمایش نمودار در مرورگر.
- meg--: آدرس فایل ذخیره سازی meg برای استفاده در نرم‌افزار Mega.
- algorithm--: الگوریتم مورد استفاده برای ساخت درخت فیلوژنتیک که میتواند مقادیر UPGMA و NJ را داشته باشد که NJ به معنای الگوریتم Neighbor Joining است.
- verbose--: در صورتی که ۱ باشد، اطلاعاتی در هنگام اجرا نیز نمایش داده می‌شود.

CHROMEISTER:

برای اجرای الگوریتم CHROMEISTER از طریق console، باید از فایل CHROMEISTER.py استفاده کنید. این فایل ابتدا آرگومان‌ها را پردازش کرده و سپس الگوریتم را با پارامترهای داده شده، اجرا می‌کند.

آرگومان‌های ورودی به شرح زیر می‌باشند (وارد کردن آرگومان‌های قرمز اجباری می‌باشد. آرگومان‌های سبز با توجه به ایده‌های خودمان اضافه شده است. آرگومان‌های بنفش مربوط به فاز ۳ پروژه می‌باشد.):

- h-: نمایش پیغام راهنمایی در مورد آرگومان‌های ورودی
- db--: آدرس فایل fasta ژنوم Database. k-merهای یکتا و بدون همپوشانی از این ژنوم استخراج می‌شود. در نمودار dot-plot این ژنوم بر روی محور X قرار می‌گیرد.
- query--: آدرس فایل fasta ژنوم Query. k-merها و مکمل معکوس (Reverse Complement) آن‌ها از این ژنوم استخراج شده و در k-merهای دیتابیس به صورت غیر دقیق به دنبال آن‌ها گشته می‌شود. در نمودار dot-plot این ژنوم بر روی محور Y قرار می‌گیرد.
- db-name--: نام مورد استفاده برای Database به جای نام فایل در خروجی‌ها.
- query-name--: نام مورد استفاده برای Query به جای نام فایل در خروجی‌ها.
- kmer-len--: طول k-merها.
- kmer-key-len--: طول کلید k-merها که به صورت دقیق مقایسه می‌شود.

- --hash: تعیین نحوه‌ی استفاده از نوکلئوتیدها در محاسبه‌ی هش. اگر مقدار "z-based" که پیش‌فرض نیز هست وارد شود، با استفاده از z و فرمول مقاله هش محاسبه میشود. برای یک تابع هش دلخواه باید آن را به صورت دنباله‌ای از * و 1 و به طول k-mer وارد کرد که در آن 1ها به معنی نوکلئوتیدهایی است که در محاسبه‌ی هش دخیل‌اند و * به معنی آن است که در این نقطه مقدار بدون توجه به نوکلئوتید، صفر است. برای مثال برای k-mer به طول ۱۰، رشته‌ی روبرو فقط ۳ نوکلئوتید از ابتدا و انتها را در محاسبه‌ی هش دخیل می‌کند: 111****111
- --z: مقدار z در محاسبه‌ی hash در فرمول مقاله.
- --dimension: اندازه‌ی طول و عرض ماتریس برخورد و dot-plot.
- --out-dir: آدرس فولدر جهت ذخیره‌ی خروجی‌ها.
- --filter-mode: نحوه‌ی فیلتر کردن k-merها برای ساختن hit matrix. اگر مقدار "one" باشد، تنها از hitهای با count == 1 استفاده می‌شود. اگر مقدار "min" باشد، ابتدا کمینه تعداد تکرار را از تمام مقادیر کم کرده و سپس کمینه مقادیر جدید به جای یک بار تکرار، انتخاب می‌شود.
- --diag-len: میزان گسترش نقاط در قطرها.
- --neighbour-dist: فاصله‌ی نقاط تا نقطه‌ی ماکسیمم در سطرها و ستون‌ها که در صورت داشتن مقدار، حذف نمی‌شوند. (در قسمت ایده‌ها توضیح داده خواهد شد)
- --kernel-width: طول و عرض کرنلی که در آن بررسی می‌شود یک نقطه باید به عنوان نقطه‌ی پرت، حذف شود یا بماند برابر دو برابر این مقدار به اضافه‌ی یک می‌باشد. (در قسمت ایده‌ها توضیح داده خواهد شد)
- --dist-th: آستانه فاصله‌ی استفاده شده در محاسبه امتیاز.
- --omit-lsgrs: اگر این آرگومان وارد شود، LSGRsها جستجو نمی‌شوند.
- --sampling-value: ضریب کوچک کردن dot-plot برای اینکه HSPها بهتر پیدا شوند.
- --diag-separation: آستانه‌ی فاصله از قطر اصلی برای تشخیص نوع رویدادها (Type of Events).
- --hsp-th: حداقل اندازه‌ی HSP

- save-output--: برای تعیین اینکه چه خروجی‌هایی ذخیره شوند:
  - m: hit_matrix.mat
  - d: hits-XY.hits
  - p: نمودارها با فرمت png
  - H: نمودارها با فرمت html
  - L: events.txt
  - h: fastas_headers.csv
  - s: score.txt
- verbose--: در صورتی که ۱ باشد، اطلاعاتی در هنگام اجرا نیز نمایش داده می‌شود.

ساخت فایل متا:

فایل متا باید شامل اطلاعاتی از فایل‌های مورد استفاده در تولید ماتریس فاصله باشد. برای ساخت این فایل از روی فایل‌های fasta کدهایی زده شده با نام‌های زیر و در خروجی قرار داده شده:

- create_meta_corona.py
- create_meta_ebola.py
- create_meta_influenza.py
- create_meta_mito.py

نتایج فاز سوم:

با توجه به شماره‌ی دانشجویی اعضای گروه، شماره‌ی یک باید بررسی میشد.

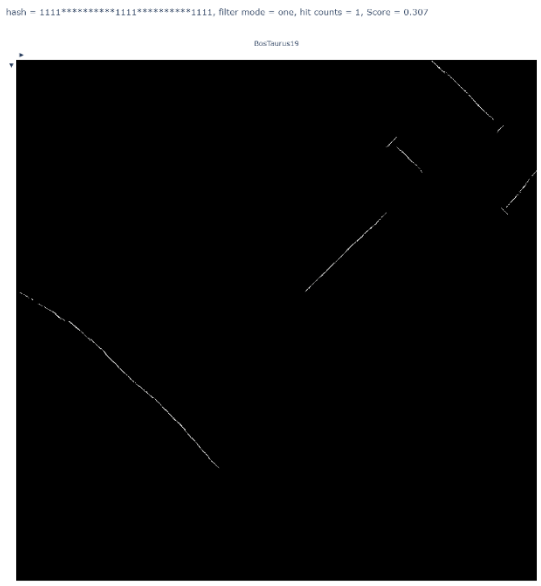
هدف از این آزمایش این بوده که آیا با تغییر تابع هش به صورت‌های گفته شده همچنان نواحی حفاظت شده، با تعداد برخورد یک مشاهده میشوند یا تعداد برخوردها افزایش می‌یابد.

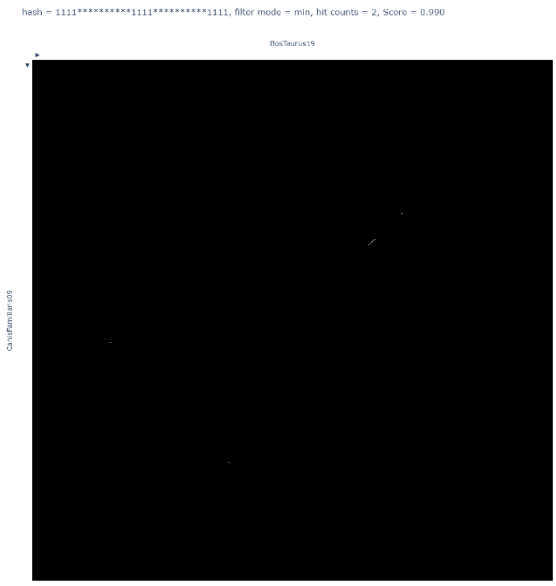
برای همین برای هر تابع هش گفته شده آزمایش را دوبار تکرار کردیم. یکبار تنها با نگه‌داشتن برخوردهای یکتا مانند مقاله‌ی CHROMEISTER و بار دیگر با نگه داشتن برخوردها با شرایط گفته شده در آزمایش که تعداد برخوردها برای حفظ آن برخورد در عنوان نمودار با عبارت "hit counts = n" بیان شده است.

همانطور که در نمودارها مشاهده می‌شود، همچنان نواحی حفاظت شده با "hit counts = 1" مشخص هستند.

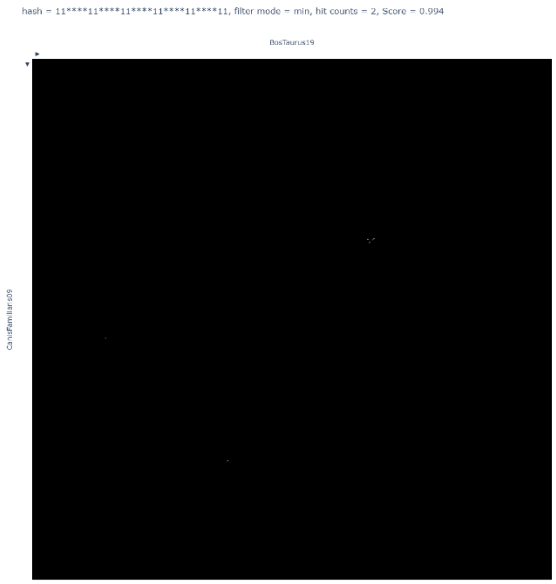
از نظر حافظه هیچ تفاوتی و از نظر زمان تفاوت چندانی با فاز دوم ندارد.

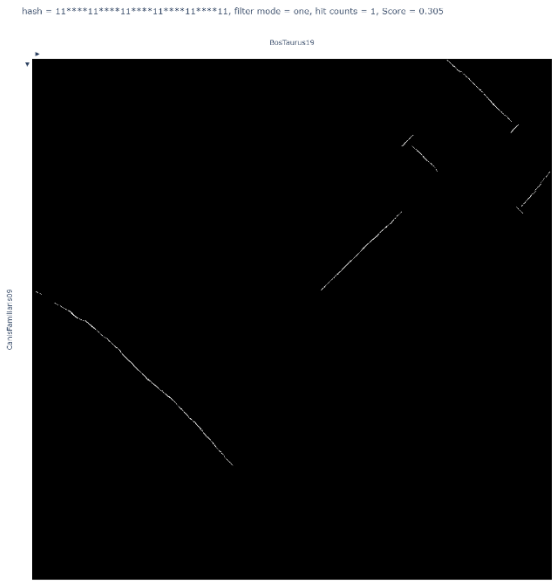
تابع هش 1111**********1111**********1111:



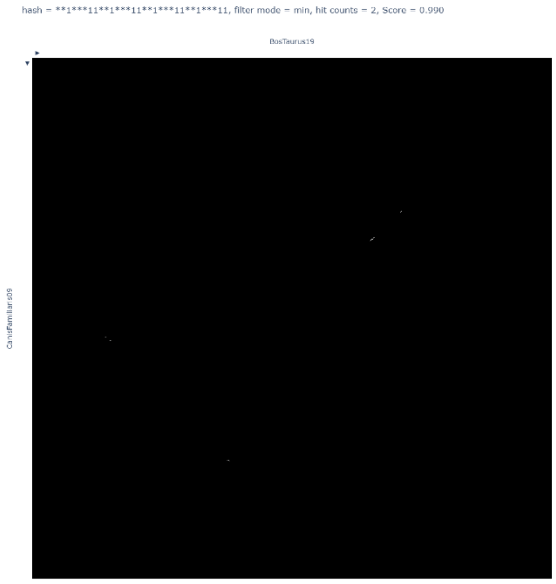


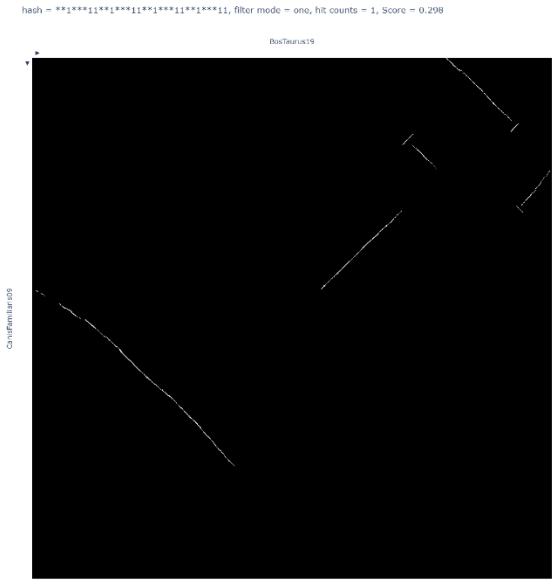
تابع هش 11****11****11****11****11****11:





تابع هش 11***1**11***1**11***1**11***1**:





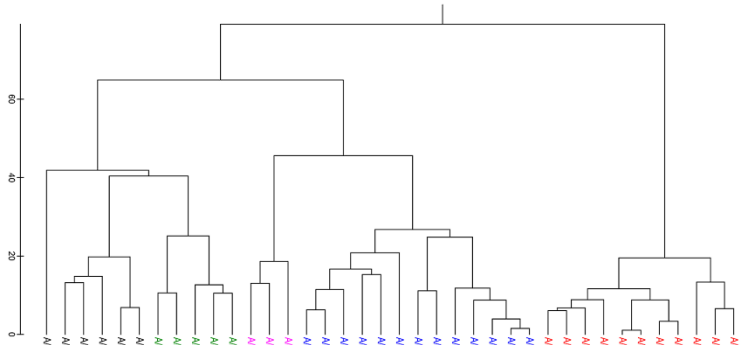
نتایج ساخت درخت فیلوژنتیک:

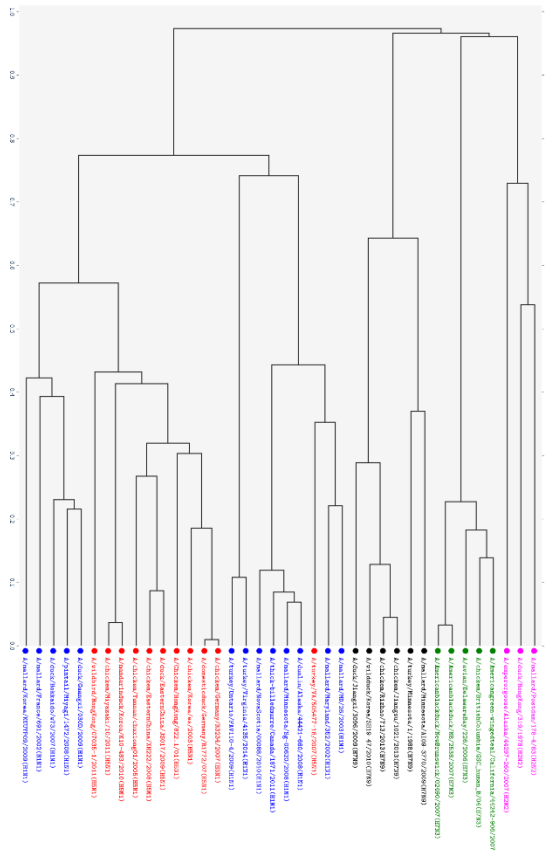
برای ساخت درخت فیلوژنتیک با استفاده از کد phlogeny.py، ابتدا ماتریس فاصله با استفاده از score در الگوریتم CHROMEISTER بدست می‌آید. چون الگوریتم CHROMEISTER ژنوم‌ها را در ۲ حالت database و query دریافت میکند، score هر دو حالت محاسبه شده و مقدار مینیمم به عنوان فاصله‌ی دو ژنوم در نظر گرفته میشود.

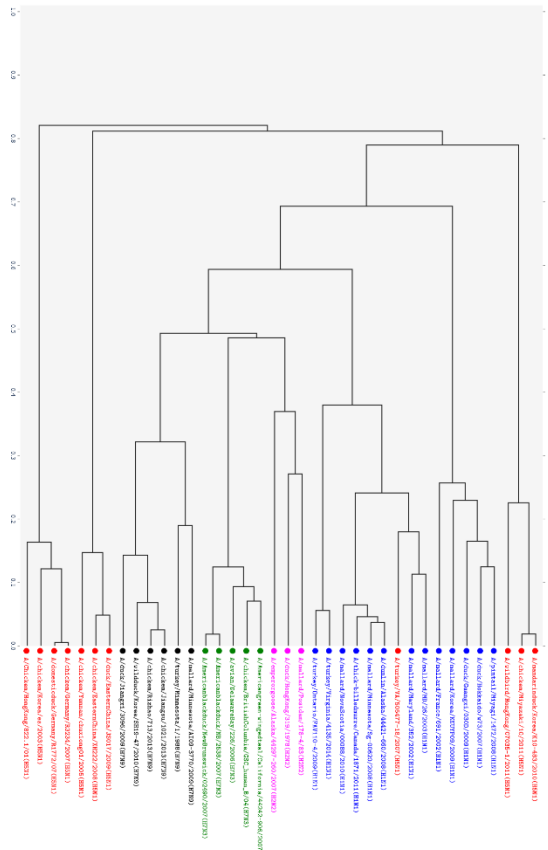
سپس با استفاده از این ماتریس و ۲ روش UPGMA و Neighbor Joining درخت فیلوژنی را بدست آوردیم. برای مقایسه، عکس همین درخت‌ها از مقاله‌ی Li, Y., He, L., Lucy He, R. *et al.* A novel fast vector method for genetic sequence comparison. *Sci Rep* **7,** 12226 (2017). نیز آورده شده است.

در هر بخش ابتدا عکس مقاله آورده شده، سپس دو عکس مربوط به کد ما (عکس سمت راست UPGMA و عکس سمت چپ Neighbor Joining).

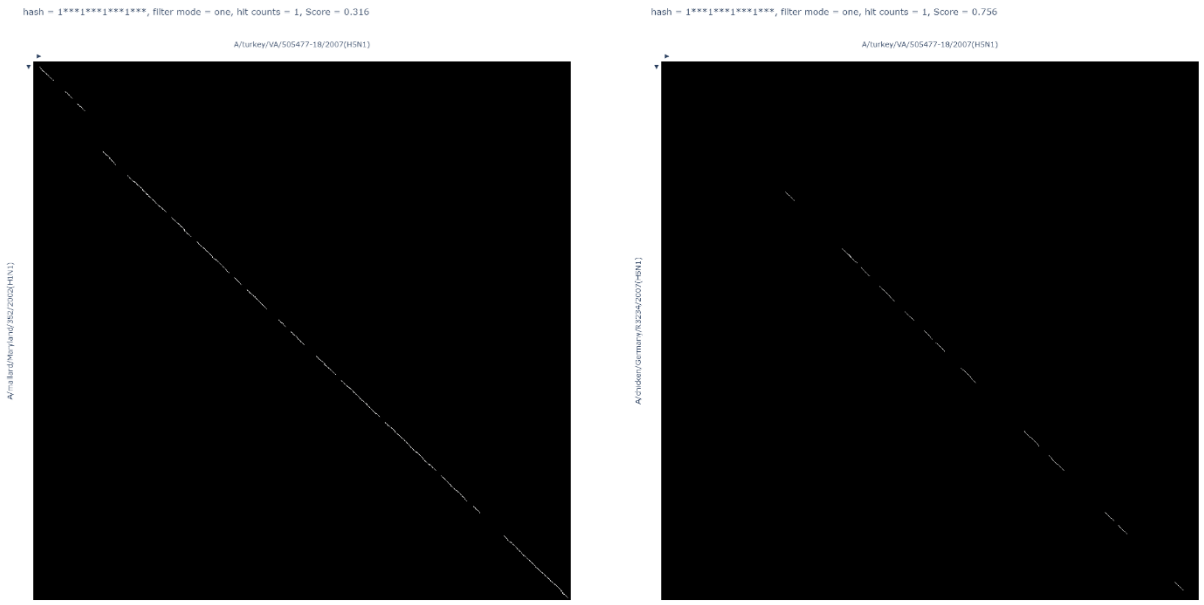
۳۸ ویروس آنفولانزای نوع A:



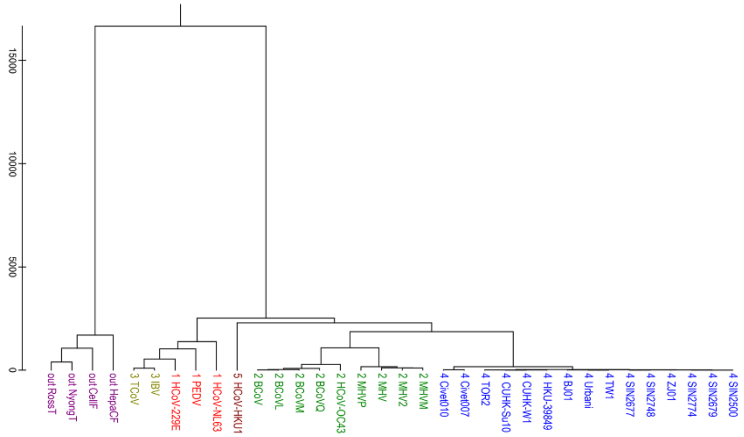


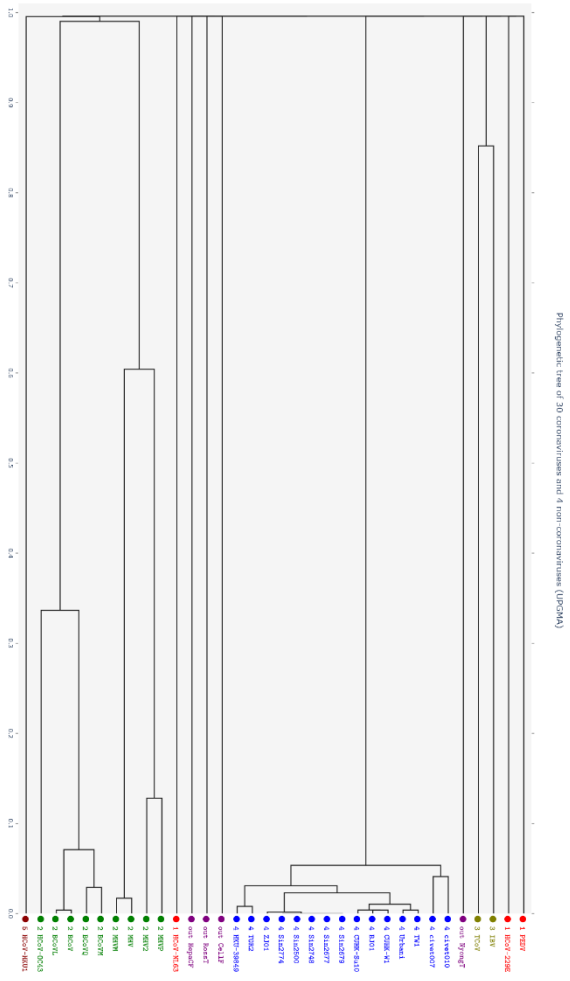


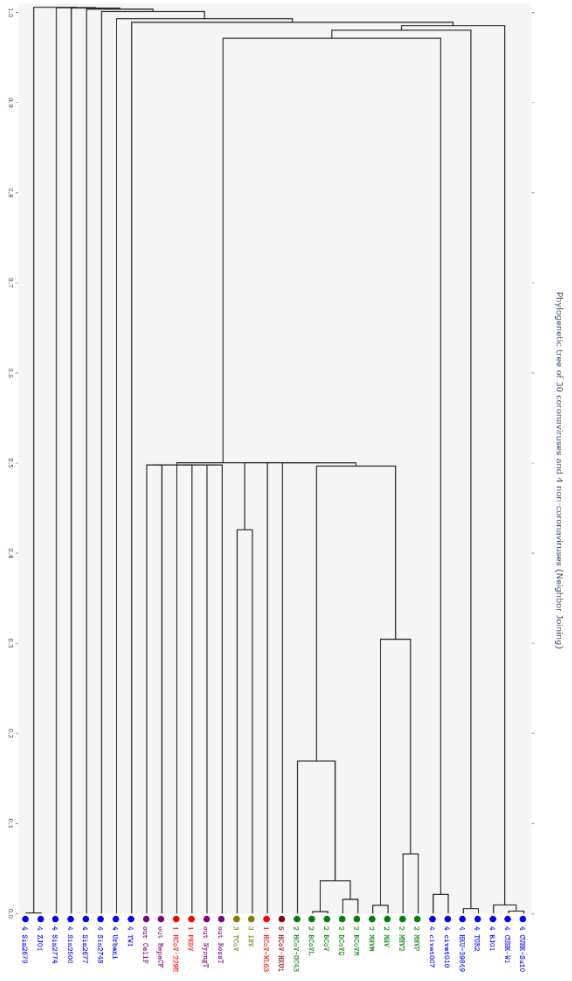
چون روش امتیاز دهی ما مبتنی بر alignment بوده و ژنوم‌های استفاده شده فقط برای قسمت N ویروس بوده‌اند، بعضی از ویروس‌های H5N1 و H1N1 جایگاهشان با مقاله متفاوت است. برای بررسی دقیق‌تر این موضوع، برای مثال، dot-plot مربوط به ۳ ژنوم (۲تا H5N1 و یکی H1N1) را کشیده‌ایم و همانطور که دیده میشود، یکی از H5N1ها به H1N1 مشابه‌تر است تا H5N1 دیگر که به دلیل این است که فقط ژنوم قسمت N ویروس‌ها مقایسه شده است:



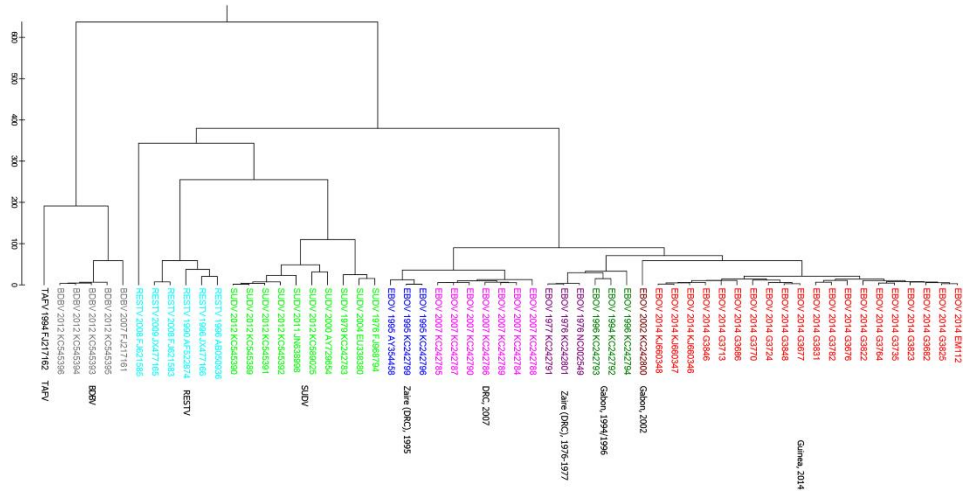
۳۰ ویروس کرونا و ۴ ویروس غیر کرونا:

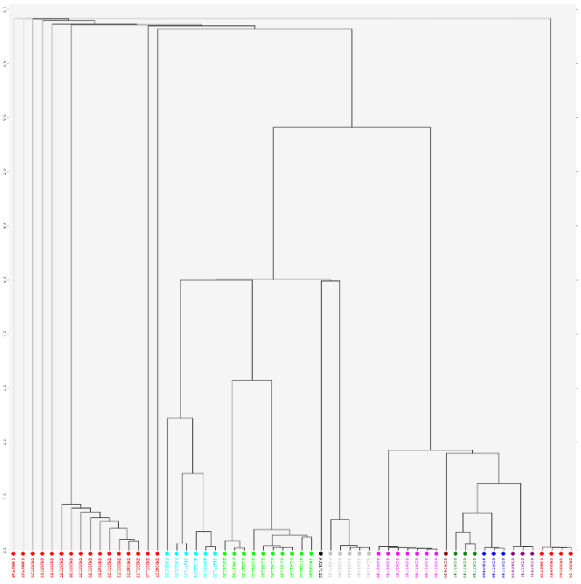


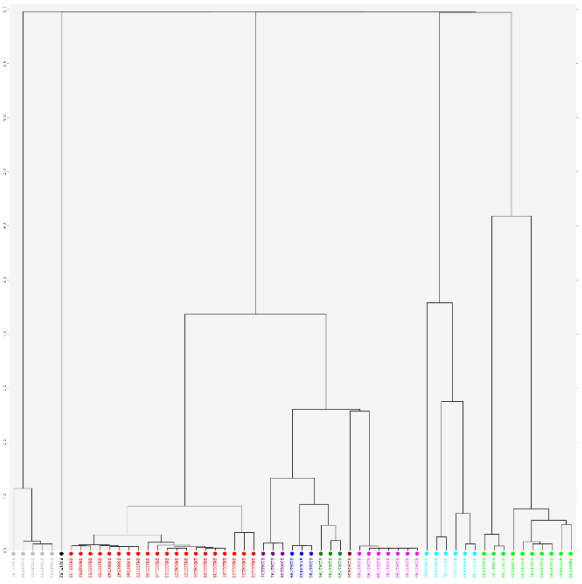




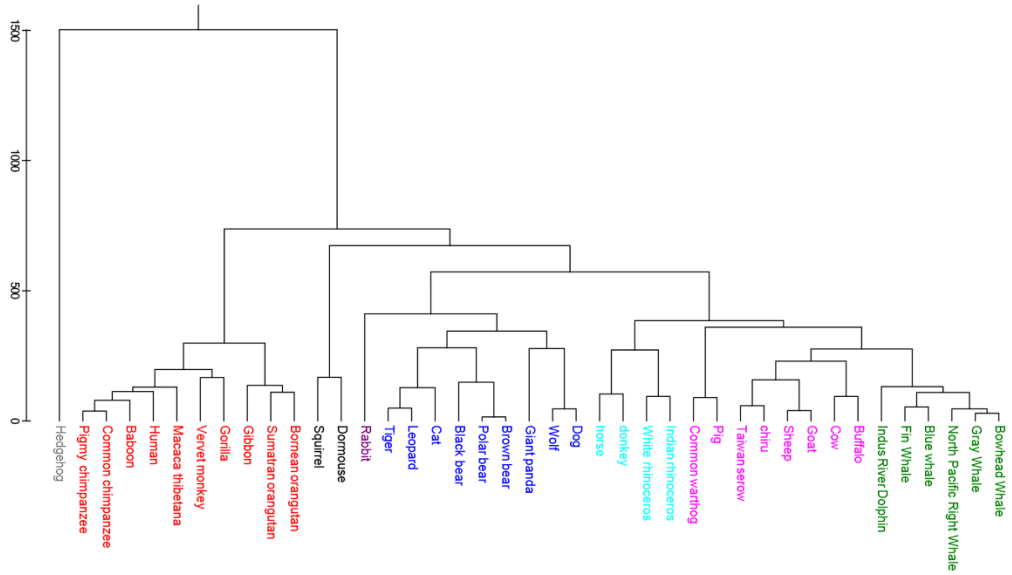
۵۹ ویروس در سرده ابولا–ویروس:

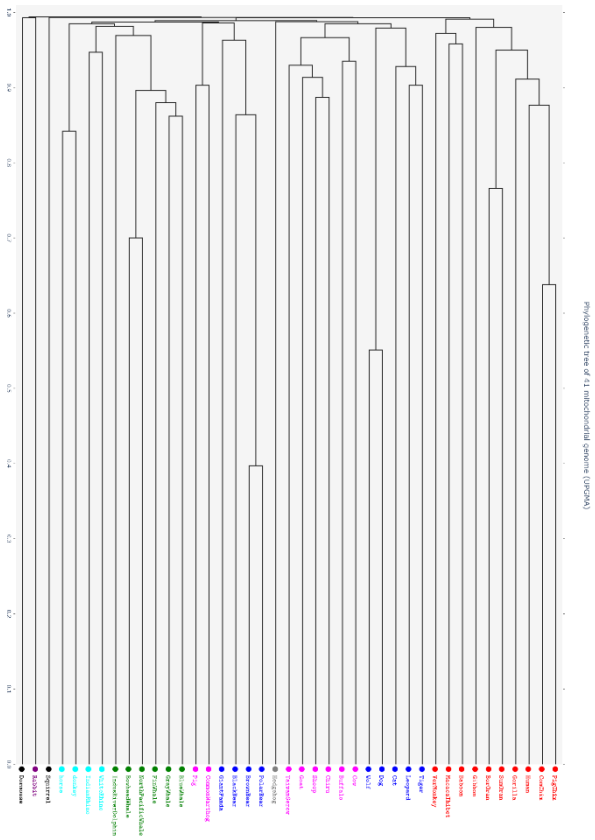


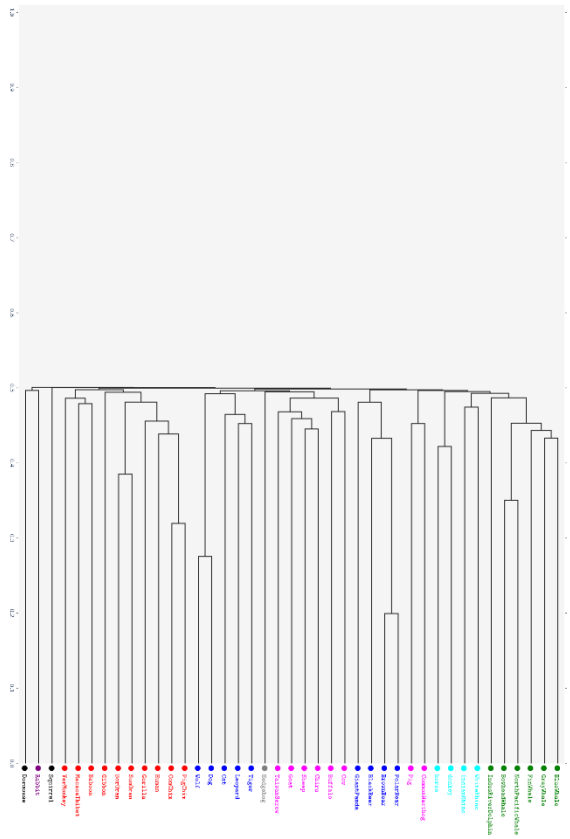




۴۱ ژنوم میتوکندریایی:







ایده‌ها:

در بخش پیاده سازی CHROMEISTER ایده‌هایی زده شده بود که در فاز دوم بیان شده بود.

ساختار فایل‌های فرستاده شده:

- CHROMEISTER.py: پیاده سازی الگوریتم CHROMEISTER.
- phylogeny.py: کد ایجاد درخت فیلوژنتیک.
- requirements.txt: پکیج‌های پایتون مورد نیاز برای اجرای کدها.
- phylogeny outputs: فایل‌های تولید شده پس از اجرای phylogeny.py و فایل‌های meta استفاده شده.
- create meta code: کدهای مورد استفاده برای ساخت فایل meta.json.