Vírusok gazdatestének klasszifikációja

Csatári Jakab

2022

Bevezetés

A projektmunkámnak szerettem volna valamilyen bioinformatikához kapcsolódó feladatot választani. Az NCBI oldalán rengeteg DNS szekvenciát lehet találni, ezek közül a vírusok voltak szimpatikusak. Hét különböző vírus rendelkezésre álló DNS-éből kiindulva a projektem célja, hogy minél nagyobb pontossággal megjósoljam, hogy a szekvenált vírus emberi gazdatestből származik-e, vagy sem.

Nem vagyok túlságosan jártas bioinformatikából, viszont próbáltam utánaolvasni, és nem találtam ismert módszert, amivel ez egyértelműen meghatározható - de ha mégis van ilyen, akkor is érdekes lehet a mélytanulásos megközelítésből vizsgálni. Különböző modelleket vizsgáltam (LSTM, Bi-LSTM, GRU, 1DConv) egyféle vírusra tanítva, majd megpróbálkoztam, azzal, hogy ezek eredményét aggregáljam össze.

A vizsgált adathalmaz

A felhasznált vírusszekvenciákat az NCBI Virus-ról (www.ncbi.nlm.nih.gov/labs/virus/vssi/) töltöttem le, de a teljes adathalmazt feltöltöttem a Kaggle-re, itt megtalálható:
https://www.kaggle.com/datasets/jakabcsatri/viruses-ncbi

Az adathalmaz fasta formátumban tárolt DNS szekvenciákat és hozzájuk tartozó metaadatokat tartalmaz csv formátumban. Természetesen a metaadatok között szerepel a 'Host' is, ahol vagy Homo Sapiens áll, vagy más faj, esetleg üres.

Első lépésben az adatot tisztítottam, mivel üres Host mező jelenthet bármit - az NCBI dokumentációja alapján azt jelenti, hogy a feltöltött szekvenciához egyszerűen nincs megadva a gazdatest - ezért ezekkel az adatokkal nem fogunk tudni dolgozni. Továbbá ismert, hogy egy DNS-ben négy nukleotid szerepelhet (adenin, citozin, guanin és timin), egy tökéletesen szekvenált DNS-ben tehát csupán A, C, G, T karakterek szerepelhetnek. A gyakorlatban nem mindig sikerül tökéletesen, vannak olyan szekvenciák, amiben egy-egy nukleotid, vagy egy hosszabb részlet ismeretlen, csak részlegesen lett szekvenálva. Az ilyen adatokat is kiszűrtem.

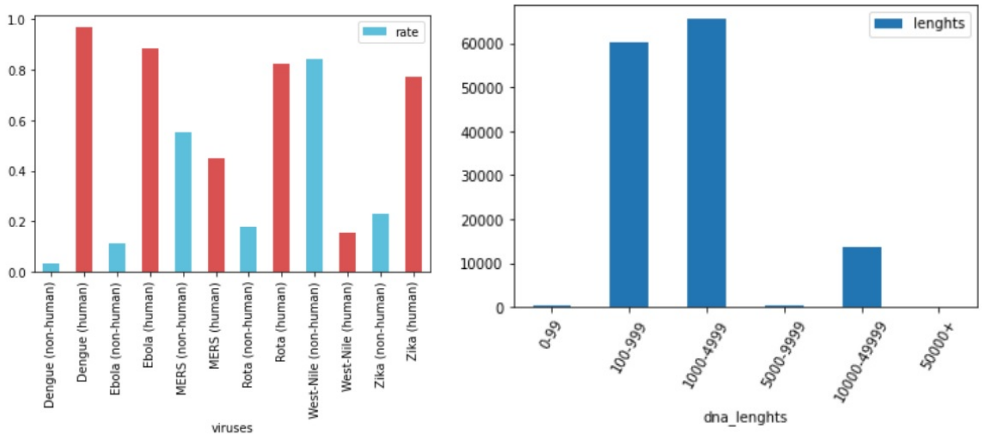
```
.....GTATTGGAGTTACAGGCGTTATAATTGCAGTTATCGCTTTATTCTGTATATGCAA
ATTTGTCTTTTAGTTTTTCTTCAGATTGCTTCATAGCAAAGCTCAGCCTCAAATCAATGA
AACCAGGATTTAATTATATGGATTACTTGAATCTAAGATTACTTGACNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNCGACATCAATCTAGTTATCTCTTTGAGAATGATAAACTTGATGAAGA
TTAAGAAAAAGGTAATCTTTCGATTATCTTTAATCTTCATCCTCGATTCTACAATCA...
...
```

Részlegesen szekvenált vírus részlet

Az adattisztítás után megmaradt adatok eloszlása vírus szerint:

Vírus	Adattisztítás előtt	Adattisztítás után
Dengue	40701	30920
Ebola	3993	2543
MERS koronavírus	1580	1301
Influenza	985050	861150
Rota vírus	111446	99730
Nyugat-nílusi vírus	22127	4650
Zika	2326	1296

Itt már látszik, hogy az influenzából megmaradt adat igen sok. Tekintve hogy egy vírus szekvencia tipikusan pár tízezer karakter hosszú, elég reménytelennek tűnik ezen a ponton, hogy az influenza adataira alkalmazzak neurális hálókat - a Google Colab által biztosított számítási kapacitás mértéke és/vagy a tanításhoz szükséges ráfordított idő hiánya miatt is. Ezért úgy döntöttem, az influenza tanítását kihagyom, de a validációnál hasznos lehet, hogy olyan vírusra is teszteljük az aggregált modellünket, amire nem tanultunk rá, ezzel vizsgálva vajon más vírusokra mennyire általánosodhat.



Bal: Vírusokra lebontott emberi és nem-emberi arány.

Jobb: DNS szekvencia hosszok az egész adathalmazon.

A leghosszabb előforduló DNS string hossza 30423 karakter, illetve láthatjuk, hogy az egyes vírusokon belül az emberi és nem emberi host arány nem kiegyensúlyozott (a legszerencsésebb még a MERS esetében), ezzel is foglalkoznunk kell a tanítás során.

Előfeldolgozás

A modell tanítást Colab-on végeztem, ahhoz hogy a szükséges adatokat kényelmesen egy helyre gyűjtve tudjam kezelni, készítettem egy csv-t mely a fasta file-ból kiolvasott DNS stringeket, virus id-kat tartalmaz, illetve egy boolean flag-et, hogy a gazdatest ember-e.

	dna_string	virus_id	is_host_human
0	ACGTGGACCGACAAGAACAGTTTCGACTCGGAAGCTTGCTTAACGT...	1	True
1	ATTCGAACAGTTTCGAATCGGAAGCTTGCTTAACGTAGTTCTAACA...	1	True
2	GAACAGTTTCGAATCGGAAGCTTGCTTAACGTAGTTCTAACAGTTT...	1	True
3	ACAGTTTCGAATCGGAAGCTTGCTTAACGTAGTTCTAACAGTTTTT...	1	True
4	ACGATAAGAACAGTTTCGAATCGGAAGCTTGCTTAACGTAGTTCTG...	1	True
...	...	...	...
140435	GTTGTTGATCTGTGTGAATCAGACTGCGACAGTTCGAGTTTGAAGC...	6	False
140436	GCAATATTTGAAGAGGAAAAAGAGTGGAAGACTGCAGTGGAAGCTG...	6	True
140437	ATGAAAAACCCAAAGAAGAAATCCGGAGGATTCCGGATTGTCAATA...	6	False
140438	ATGAAAAACCCAAAGAAGAAATCCGGAGGATTCCGGATTGTCAATA...	6	True
140439	AGTTGTTGATCTGTGTGAGTCAGACTGCGACAGTTCGAGTCTGAAG...	6	False

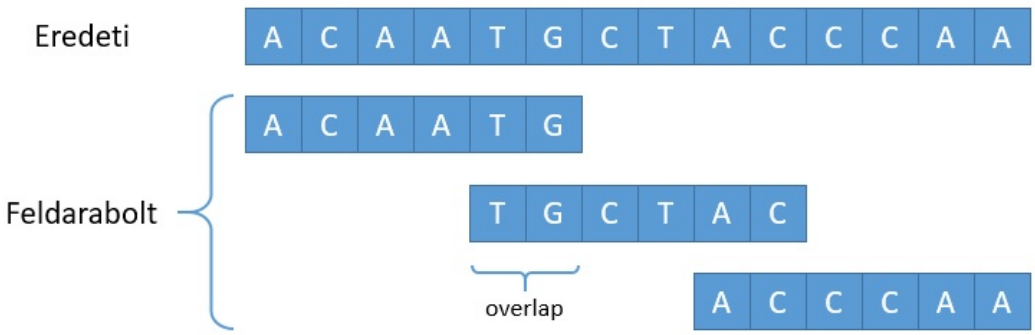
A kiindulásnak használt csv

(A vírus id-k: 1: Dengue, 2: Ebola, 3: Mers, 4: Rota, 5: West-Nile, 6: Zika)

Az egyik legfontosabb kérdés az előfeldolgozás során, hogy hogyan alakítsuk a szekvenciákat input tensor-okká - karakterenként tokenizáljunk, esetleg szavakként tekinitsünk-e rájuk. Az én megközelítésem az volt, hogy állítható split hosszt és overlap értéket megadva úgy bontottam fel a szöveget, hogy minden részlet éppen olyan hosszú legyen, mint a megadott split hossz és a következő annyi karaktert tartalmazzon az előzőből, amennyi az overlap érték.

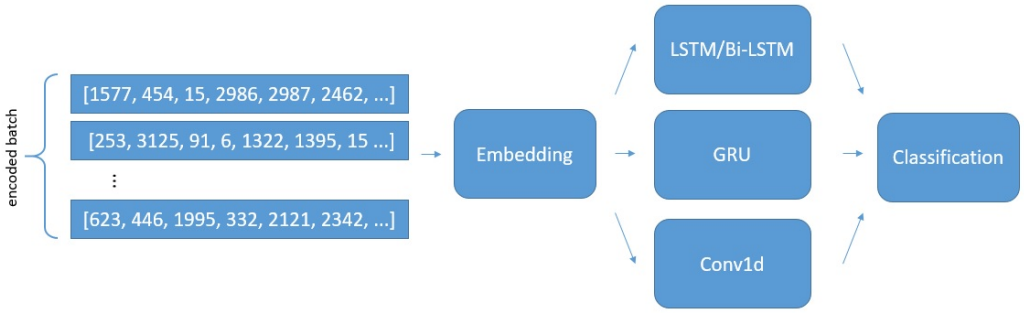
Tehát például a karakter szerinti felosztás éppen a split_len = 1, overlap = 0 -nak felel meg, vagy a bioinformatikában használt k-mer fogalma a split_len = k, overlap = k-1. Próbálkozások során azt tapasztaltam, hogy a 6-hosszú 2-átfedéses verzió teljesített legjobban a tanítás során.

Ha például a DNS stringünk "ACAATGCTACCCAA", akkor így az ["ACAATG", "TGCTAC", "ACCCAA"]-t kapjuk:



A használt modellek

A word-tokenizer, amit használtam az előforduló rész-szekvenciákat számozza meg, majd az így kapott vektorokat Embedding layer-rel one-hot vektorokként fogjuk betáplálni a neurális hálóinkba. Az modellek implementációihoz PyTorch-ot használtam.



Folyamatábra a (pre-aggregációs) modellekhez

A rendelkezésre álló adatokat szétválasztottam vírus szerint, és mindegyiket külön-külön training, test és validációs halmazra osztottam 80-10-10 arányban. A fenti modellel egyszerre csak egy vírust tanítottam, így minden vírushoz különböző modell és vagy hiperparaméterek választhatók, amelyik arra a legjobban teljesít. Ezeket fogjuk egy aggregált modelbe betölteni második lépésben.

Vírusonkénti tanítás

Három metrikát vettem figyelembe a tanítás során. Elsődlegesen az accuracy-t vizsgáltam, de igyekeztem elkerülni az accuracy paradoxont. Emellett természetesen a veszteség függvény csökkenését, illetve a confusion mátrixokat néztem.

Először is szükséges volt, a training adatok kiegyensúlyozása, a korábbi ábrán látszott, hogy voltak kifejezetten kiugróan kiegyensúlyozatlan vírusok (például a dengue-nél az emberi minta jóval magasabb, mint a nem emberi, de van olyan is ahol épp fordított a helyzet: nyugat-nílusi vírus). Ezen egyszerűen random undersamplinggel igyekeztem felülkerekedni, ugyan adott esetben így jóval kevesebb a tanító adatunk - mégis egész jó pontosságokat lehetett elérni ennek ellenére egy-egy vírusnál (pl. ebola, ahol szintén az emberi minta jóval több).

Vírus	Kiegyensúlyozatlan tanító adat db	Undersampling után db
Dengue	24736	1621
Ebola	2034	494
MERS koronavírus	1040	952
Rota vírus	79784	29785
Nyugat-nílusi vírus	3720	1213
Zika	1036	504

Adatelhagyás mértéke (a nagyobb halmazon) random undersamplinggel

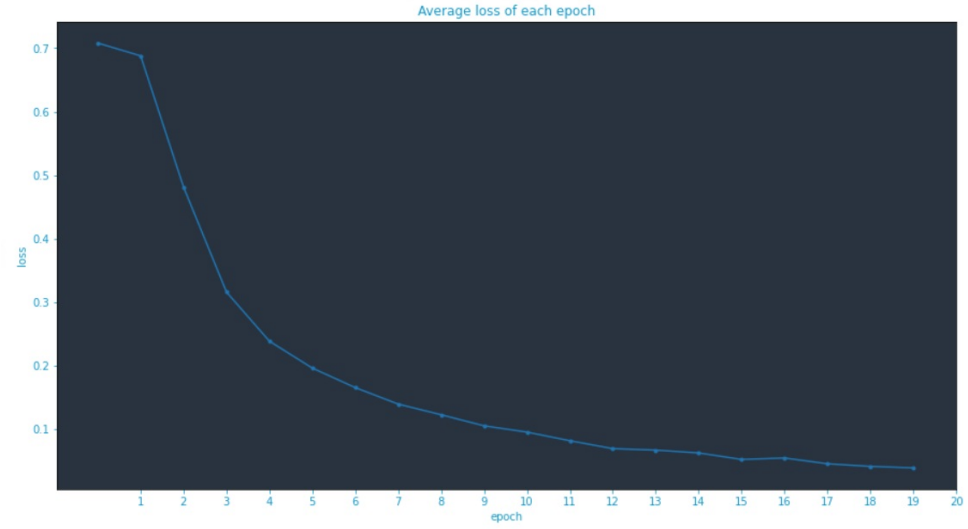
Érdekes, hogy az intuícióm azt sugallta, elsősorban a modellek teljesítménye között lesz nagy különbség és arra számítottam, hogy általában a Conv1D lesz a leggyengébb, majd követi őt a GRU, LSTM végül Bi-LSTM. Arra nem számítottam hogy jelentős különbség lesz a vírusok között, hogy milyen pontosan tudjuk a modelleinkkel klasszifikálni. Ehhez képest a modellek között viszonylag kisebb különbség volt, ellenben egy-egy vírusra jelentősen pontosabb klasszifikációt tudtunk elérni, mint egy-egy másikra.

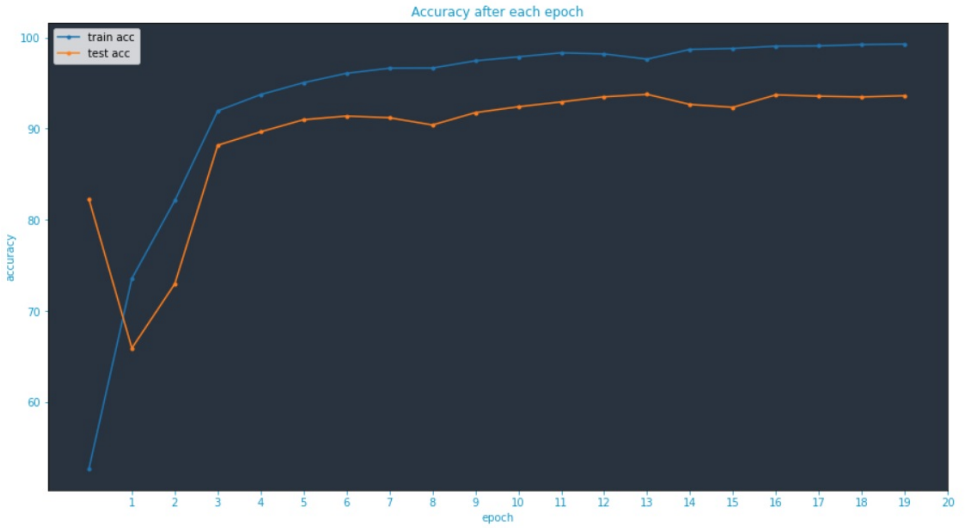
A tanítás során Adam opimizerrel és CrossEntropyLoss veszteségfüggvénnyel dolgoztam.

Tanítás eredményei

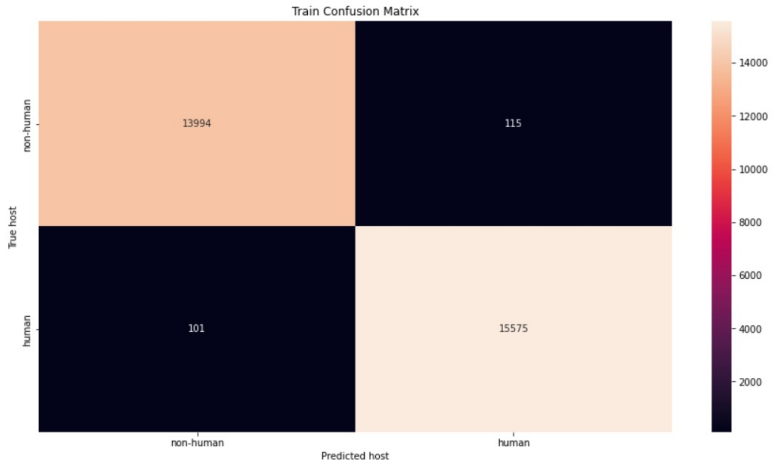
Mind a hat betanított vírushoz 4-4 modellt próbáltam minél jobban tanítani, a legjobbnak ítélt hiperparaméterrekkel a futtatások, és a hozzájuk tartozó chart-ok megtalálhatók a mellékelt notebookban, tekintsünk csak egy példa futtatást: a GRU modell tanítását a Rotavírusra.

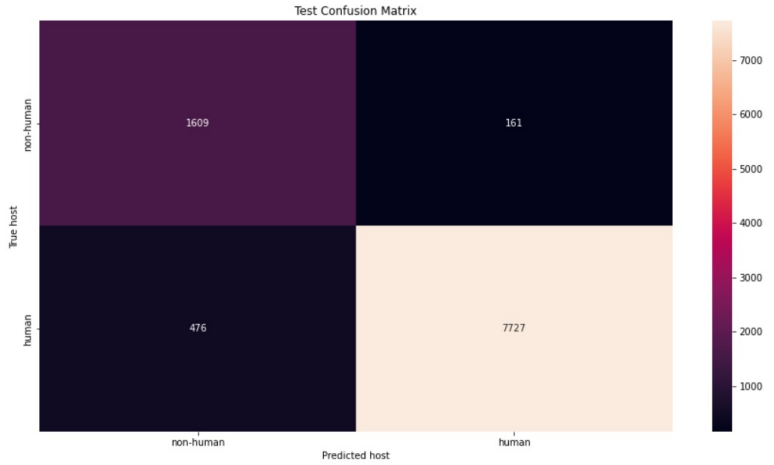
A veszteségfüggvény és a pontosság alakulása a tanítás során.





A tanítás végén a confusion mátrixok:





A legjobb modellek pontosság szerinti összehasonlítása:

	LSTM	Bi-LSTM	GRU	Conv1D
Dengue	83.93	86.58	84.41	89.55
Ebola	98.03	99.21	98.43	97.24
MERS	82.31	79.23	78.46	73.85
Rota	92.64	92.97	93.61	90.64
West-Nile	90.11	88.82	90.11	84.73
Zika	88.46	85.38	83.08	86.92

Accuracy teszt adatokon.

Mint említettem, az előzetes elvárásom az volt, hogy Conv1D $<$ GRU $<$ LSTM $<$ Bi-LSTM lesz az eredmény szinte minden vírusra. Azonban érdekes, hogy legfeljebb némileg tendál errefele dolog, de vannak ettől a hipotézistől nagyban eltérő eredmények, például a dengue-re a Conv1D adta a legpontosabb klasszifikációt, a Zika-ra is relatíven pontosabbnak számított a Conv1D, míg a Bi-LSTM általában jól teljesített, de csak az ebolánál volt a legpontosabb.

Illetve az derült ki a tanítás során, hogy az egyes vírusok adathalmazai közt jelentős különbség van, a MERS és zika vírusoknál sokkal nehezebb volt magasabb pontosságú futtatásokat csinálni, mely a végeredményen is látszik, míg például az ebola esetén a hiperparaméterek állítgatásával alig lehetett elrontani, hogy ne legyen nagyon kedvező a klasszifikáció pontossága.

Overfittinget nem tapasztaltam egyik tanítás során sem, ebben valószínűleg nagy szerepet játszott, hogy jellemzően 0.2-es dropout értékkel tanult minden modell.

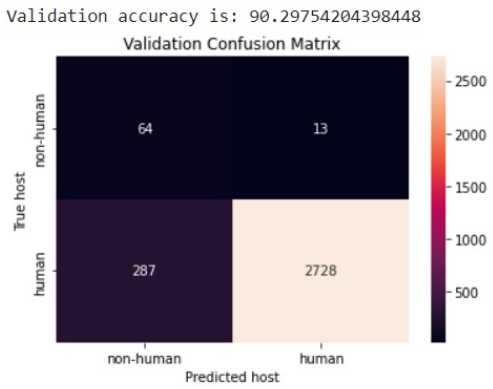
Vírusonként a legjobb modellek hiperparaméterei:

	Dengue Conv1D	Ebola Bi-LSTM	MERS LSTM	Rota GRU	West-Nile GRU	Zika LSTM
Embedding dimension	50	100	50	60	70	50
Num layers		2	2	2	2	2
Dropout	0.2	0.2	0.2	0.2	0.2	0.2
Batch size	1	2	5	100	10	3
Epochs	15	20	10	20	20	20
Learning rate	0.005	0.001	0.005	0.001	0.001	0.001

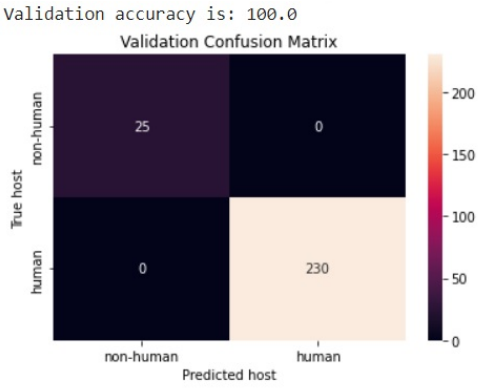
Validációs eredmények

Úgy alakult, hogy a validációs adathalmazra a legjobb modelleket lefuttatva még jobb eredményeket kaptam, mint a test adathalmazra. Ez persze nem jelent semmit, de érdekes, hogy pont így alakult mind a hat esetben. Ami kifejezetten meglepő volt, hogy az ebolát tökéletesen klasszifikálta, valahogy pont így fogtuk ki.

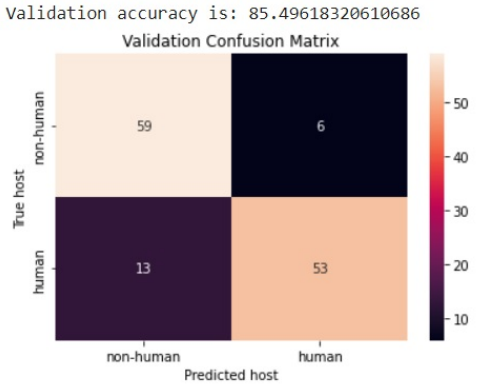
Dengue Conv1D:



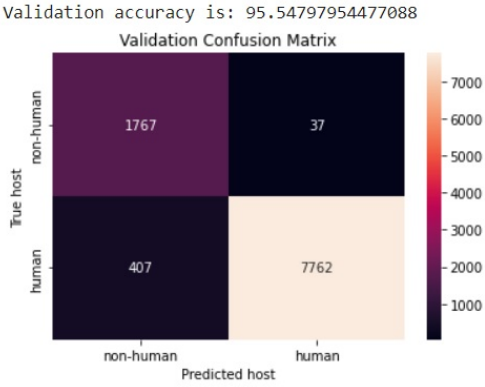
Ebola Bi-LSTM:



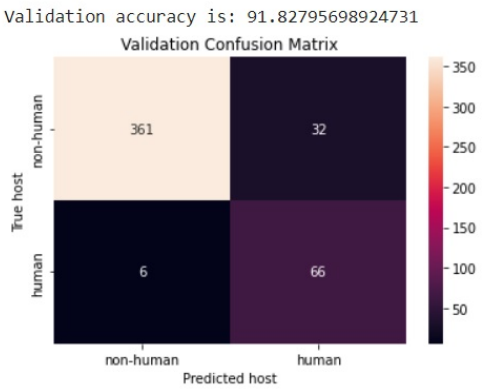
MERS LSTM:



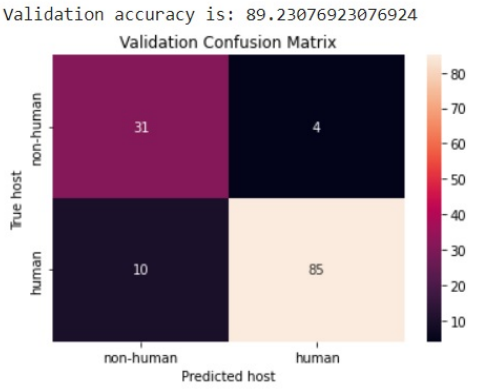
Rota GRU:



West-Nile GRU:



Zika LSTM:



Aggregált model

Miután betanítottam egyenként a vírusokat, azzal próbálkoztam, hogy olyan aggregált modellt készítsek, amely a legjobb modellek mindegyikét lefuttatja, majd az alapján ítéli meg a klasszifikáció valószínűségeket. Ezt egy érdekes ötletnek gondoltam és láttam benne potenciált.

Ehhez először az output tensorokra softmax-ot hívtam (hogy a klasszifikációs valószínűségeket megkapjam minden modellre), majd az aggregálást 3 különböző taktikával implementáltam:

1. Lineáris kombináció: a valószínűségeket egyenlően sulyozva kapjuk meg az aggregált klasszifikációs valószínűségeket (gyakorlatilag átlagolunk).

2. Maximum: A maximális valószínűségű klasszifikációt vesszük figyelembe.

3. Pozitív eltolódású max: Ha elég nagy valószínűséggel jósol egy model 1-et (hogy emberi a gazdatest), akkor azt elfogadjuk, különben max-ot veszünk.

Elméletben úgy gondolom, az átlag akkor teljesítene jól, ha minden vírus esetén a modell valami hasonló irányra tudott volna rátanulni - vagyis, ha létezik/létezne tulajdonság, amiből könnyen megmondható kézzel is, hogy egy vírus host-ja ember-e. A maximumtól azt várnám, hogy olyan vírusokra teljesítsen jól, amihez hasonlóra már jól rátanult bármelyik model. Az utolsótól úgyszintén, ott az volt az ötlet, hogy ha léteznek nagyon ellentétes vírusok, akkor esetlegesen kiszűrje a fals negatív predikciókat.

A reményeim ahhoz fűződtek, hogy valamelyik ezek közül jól fog teljesíteni teljesen ismeretlen vírusra, ezért influenzára futtattam ezt a modelt. De az eredmény nem volt túl kecsegtető; az előbbi felsorolás sorrendjében a pontosságok: 56.45, 64.9 és 40.55.

Azért szerencsére nem teljesen reménytelen a modell, bizonyos esetekben lehet értelme. Például a zikavírusnál nem sikerült túl magas pontosságot elérni (89.23), azonban aggregálva a hozzá tartozó GRU, LSTM és Bi-LSTM modelleket, a 3. változattal 91.54 pontosságot értem el, ez pedig magasabb, mint amit bármelyik modellel önállóan ki tudtam hozni.

Függelék

A felhasznált anyagok megtalálhatók a github oldalon: https://github.com/CsatariJakab/virus-host-classification

Futatás know how:

- Az adathalmaz elérhető a Kaggle-n (https://www.kaggle.com/datasets/jakabcsatri/viruses-ncbi).
- Kicsomagolás után futtatható a 'src/preprocessor.py' script, amely kifilterezi a nem hasznos adatokat és létrehozza a 'data/filtered_sequences.csv' és 'data/filtered_sequences_influenza.csv' file-okat.
- Ezután a notebookot: 'virus_host_detection.ipynb' Colab környezetben nyissuk meg
- A csv file-okat fel kell tölteni a Google Colab notebook-ba (célszerű 7z tömöríteni, a kitömörítéshez van kódrészlet a nootbook elején)
- Innentől mindent a notebookban lehet futtatni
- A betanított modellek szintén megtalálhatóak a github repoban a 'data/models/' alatt.