ADVANCED ARCITICTURE

זיו אסולין 325312999
אלון ארטמונוב 213042518
שלמה אסף 318405545

K-mer Classification of DNA Sequences

1. Introduction and Rationale

In genomics, accurately classifying DNA sequences is pivotal for discerning biological functionalities and disease-related mechanisms. Traditional methodologies, characterized by manual feature extraction and sequence alignment, are often tedious and become less effective as the size of datasets increases. This project endeavors to harness the power of machine learning for DNA sequence classification, with a focus on k-mer frequency counts. By transmuting DNA sequences into these frequency counts, we are setting the stage for a high-dimensional feature space suitable for the application of advanced machine learning techniques.

1.1 K-mer -

K-mers are subsequences of a DNA sequence of length 'k'. By converting DNA sequences into k-mer frequency counts, we can transform the problem into a high-dimensional feature space where machine learning algorithms can be applied.

The reason we chose K-Mer is as this method has been proven to be useful in DNA sequence classification in the past and has several advantages such as reducing noise, preserving local information, scalability and ease of use.

2. Objective

Our principal objective is to facilitate a streamlined, cost-effective mechanism for DNA sequence classification. We have adopted the widely recognized K-Mer Transform method to interpret the DNA sequence, subsequently converting the result into an image format suitable for Convolutional Neural Network (CNN) processing.

3. Methodology

Data Acquisition

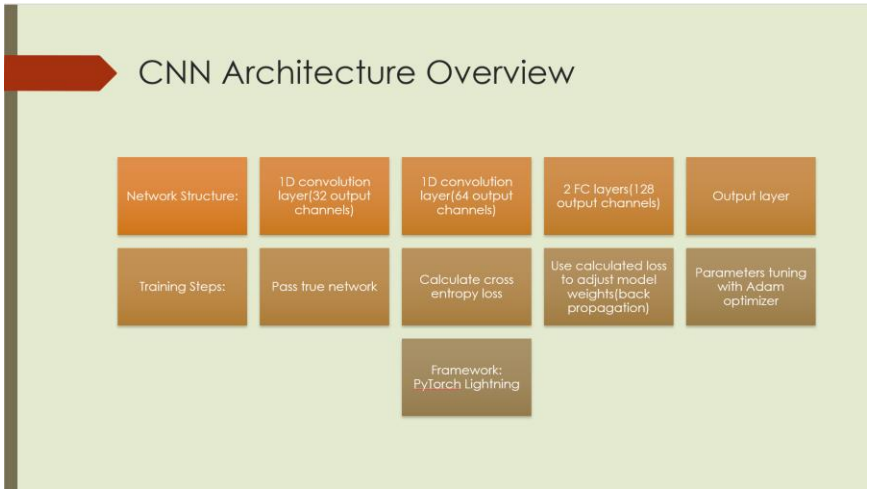
To train our model, we've developed a codebase that aggregates widely recognized lineages and their respective samples from online sources. For inference, the process can be modified to acquire the desired sample.

K-mer Analysis

K-mers are substrings extracted from DNA sequences, with each having a length 'k'. The conversion of DNA sequences into k-mer frequency counts allows us to redefine the problem in a high-dimensional framework conducive for machine learning. The preference for the K-Mer approach stems from its proven efficacy in DNA sequence classification, as well as inherent benefits like noise reduction, local information preservation, scalability, and user-friendliness.

For our CNN we have chosen to go with an Image Classification as although it may not be the optimized model to our problem it has major advantages such as having Cutting edge NN with

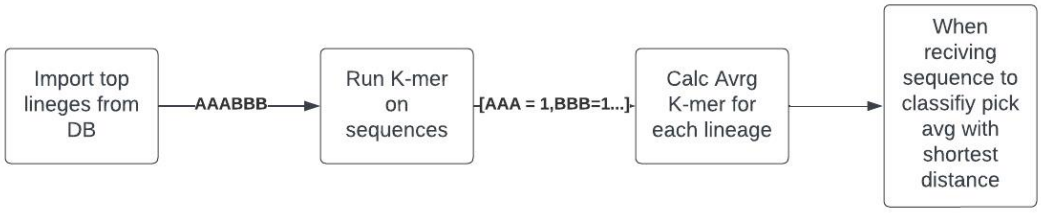
proven results and constantly improving thus by going on the “paved road” we will be able to easily maintain our code and improve it as technology advances
As shown in The Presentation Our NN Looks like this



Our Results
On the first few runs of our Model we have received amazing results With close to 89% accuracy for 100 most common lineages which almost seemed to good to be true.
Thus after further research we have discovered a bias in our training as apparently when batch parsing our sequences using K-mer the K-mer function we chose to use removed the K-mers with the over-all count of 0 from our results which caused the bias.
So after fixing our Bias we have received a more realistic but less satisfactory answer of 49% accuracy for top 10 most common lineages (compared to 100 before) thus be began looking for other ways to improve our Results

Transitioning from Deep Neural Networks (DNN) to a k-mer-based Naïve Approach:

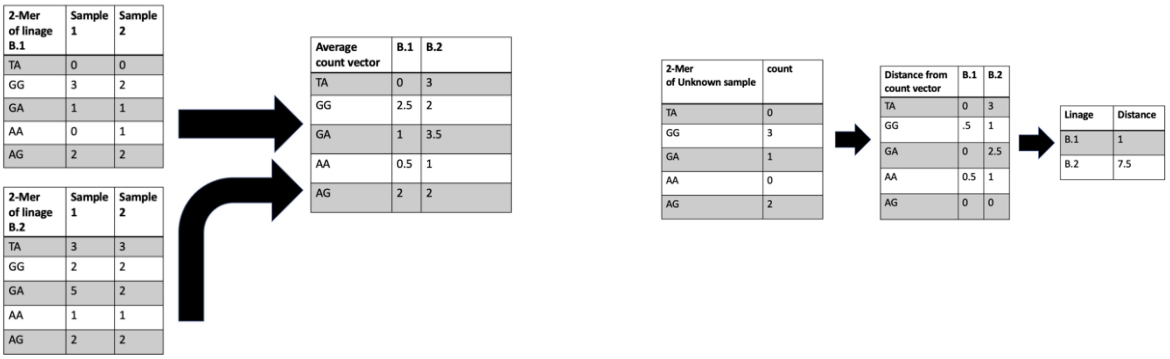
Upon analysis, it was observed that our deep neural network exhibited significant bias towards the training samples. This prompted areconsideration of alternative methods for classification. Surprisingly, a seemingly simplistic approach emerged as the most efficient. Rather than preprocessing all the samples and subsequently inputting them into the DNN, we adopted a method that involved computing an 'average k-mer count vector' for each lineage, as illustrated in figure-



For classification, the 'k-mer count vector' is computed and classification is performed based on the nearest neighbors using the Euclidean distance metric. This method proved to be highly effective, particularly in the realm of computer architecture. Notably, the classification

process involves mere addition and subtraction operations, which are computationally inexpensive.

The following tables represent classification process –



2-Mer of linage B.1	Sample 1	Sample 2
TA	0	0
GG	3	2
GA	1	1
AA	0	1
AG	2	2

2-Mer of linage B.2	Sample 1	Sample 2
TA	3	3
GG	2	2
GA	5	2
AA	1	1
AG	2	2

Average count vector	B.1	B.2
TA	0	3
GG	2.5	2
GA	1	3.5
AA	0.5	1
AG	2	2

2-Mer of Unknown sample	count
TA	0
GG	3
GA	1
AA	0
AG	2

Distance from count vector	B.1	B.2
TA	0	3
GG	.5	1
GA	0	2.5
AA	0.5	1
AG	0	0

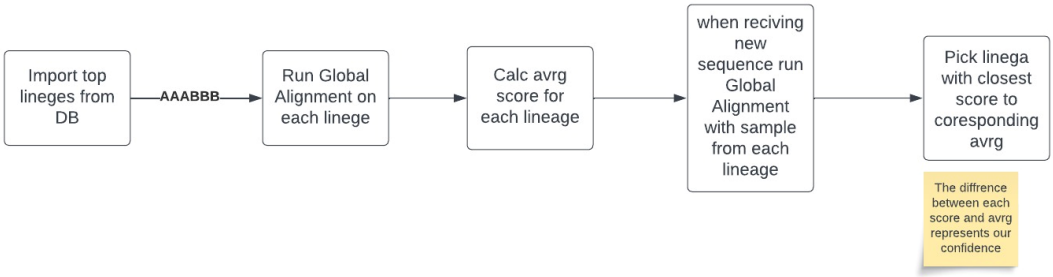
Linage	Distance
B.1	1
B.2	7.5

Result summary of the naïve approach

Top-k-accuracy	1-mer	2-mer	3-mer	4-mer	5-mer	6-mer	7-mer	8-mer
1	0.04	0.05	0.08	0.17	0.37	0.55	0.69	0.76
2	0.06	0.08	0.12	0.23	0.44	0.66	0.82	0.88
3	0.08	0.11	0.16	0.31	0.5	0.73	0.86	0.92
4	0.11	0.12	0.19	0.35	0.57	0.77	0.89	0.94
5	0.13	0.15	0.21	0.39	0.63	0.8	0.91	0.95

Global Alignment Classification

Another interesting way we tried to improve our results is by using Global Alignment.
With this algorithm in the learning phase for each Lineage class we would calculate the global alignment score of each of our samples with each other and calculate the average Global Alignment score for each Lineage.
Similarly in the Inference phase we take a sample from each linage and calculate the global alignment score and compare It to the average for a total of K global alignments where K is the number of lineages.
The flow would look something like this.

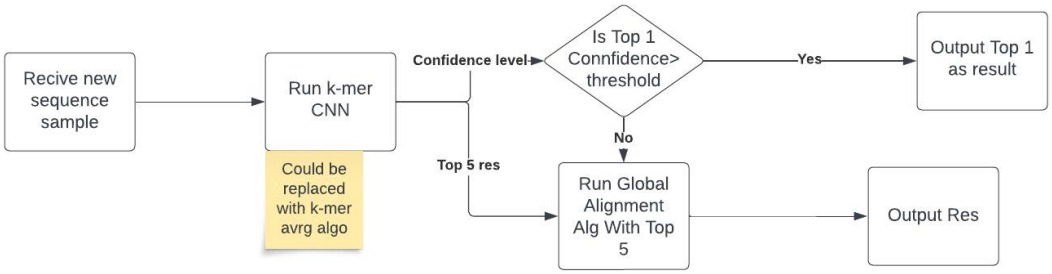


The major disadvantages of this approach is the compute time that each Run takes we even had a hard time to run the algorithm on a small data set due to the relatively long time it takes to compute the global alignment thus although we saw a promising result while working on

extremely small data set (5 lineages 2 samples from each) this method needs higher computing power in order to verify and test in a short duration.

4. Conclusions:

As unfortunately we weren't able to achieve the results we hoped for we took the time to think on how we could further improve our model and learn from our results.
The ideal design we have come up looks as the following:



In this approach we try to get the best of both worlds, in this new design when receiving a new sample we would first run the sequence through our CNN or out K-mer average distance algorithm which for it simplicity showed amazing results. Then if the answer we received was decided with a high level of confidence we would return it as the results and if the model isn't confident enough in the result it achieved then we could run our global alignment classification algorithm on the
Top 5 results which should be a lot more manageable and allow us to achieve high accuracy with a relatively low compute power.
Other option to explore is reducing the average count vector size by removing the entries with the lowest variance, this way we can greatly reduce the memory needed to support the methodology hopefully without effecting performance.