Machine Learning CS-433 - Project 2
Clustering of DNA Sequences for Primer Extraction

Younis Bouzar (younis.bouzar@epfl.ch), Derya Cögendez (derya.cogendez@epfl.ch),
Kajetan Rożej (kajetan.rozej@epfl.ch)

***Abstract*—In this project, we tackle the challenge of clustering DNA sequences for primer extraction, a crucial step in the EPFL make project Genorobotics team's effort to streamline plant classification for biodiversity research. To achieve successful primer extraction, clusters of relatively similar DNA sequences are required. We employ machine learning models for sequence embedding and subsequently apply clustering algorithms such as k-means and DBSCAN. Based a novel evaluation metric involving Clustal, our results show that autoencoder with convolutional layers yields the best results for this task. However, further exploration of deeper neural networks and understanding cluster characteristics needs to be done.**

I. Introduction and Motivation

The EPFL make project Genorobotics team is working to create a pipeline for easily classifying plants in the field for biodiversity research. There exist specific regions of DNA that, when amplified and sequenced, can be used to identify a species, this is known as DNA barcoding[KE07]. To perform the amplification, a technique called the Polymerase Chain Reaction (PCR) is used. PCR makes numerous copies of a specific DNA sequence from a small starting amount. This process is also crucial in various different scientific and medical applications like medical diagnostics (PCR-test) and forensic analysis[Wik23b]. An important step in PCR is the annealing phase. The reaction temperature is lowered to allow short DNA sequences called primers to bind to single-stranded DNA. These Primers are designed in a way such that they bind to a specific part of the DNA sequence we are interested in. Afterwards, in the extension phase, DNA polymerase synthesizes new DNA strands.

As primers attach to a specific sequence on DNA, currently they need design a set of primers specific to each plant. In a field trip, it is expected to get samples from 300-1000 plants, therefore it takes a lot of time and money to prepare a large number of primers for field research. As this type of research is usually done in remote regions, they need to prepare their primers beforehand. To streamline their pipeline, the Genorobotics team are looking to design universal primers (primers that will work on multiple plants barcode genes).

To extract primers from a set of DNA sequences there are pre-existing tools such as primalscheme[QS17]. However, to have good results from these tools, we need clusters of DNA sequences that are relatively similar. To be able to see if clusters of DNA sequences are relatively similar we have tools such as Clustal, blastN etc. The Genorobotics team wanted us to focus on Clustal, as for their experiments this gives the best results for their purpose (multiple sequence alignment). Thus, in this project we try to get clusters that have relatively low gaps after alignment using the tool Clustal. However, high percentage of gaps in a Clustal alignment does mean the clusters are not good. We provide further metrics for our clusters in section III.

In this project we train/use machine learning models to get embedding of DNA sequences, which we use for clustering with k-means, for further information on our methods please refer to the next section II. Ideally, we want to get a low number of clusters, as we explained in the previous paragraph for simplifying expeditions, from which we can extract primers.

II. Models and Methods

There are numerous ways to cluster a given data set. Algorithms like k-means and DBSCAN are very powerful, however they are unable to capture deeper patterns in the data. Our approach therefore consists on encoding the data to make it useful for neural networks, train neural networks to extract features/embed the data to then feed it to a clustering algorithm that provides the wanted results. For each step in our approach there are multiple methods to choose from. It is a priori very difficult, if not impossible, which combination of methods will perform best. We therefore selected a handful and analyze the results in Section IV.

A. Encoders

DNA sequences are typically represented using a combination of the four letters "A", "C", "G" and "T" corresponding to the four nitrogen bases. The length of these sequences vary greatly, but in our case they range from $600 - 1000$. Machine learning or deep learning models however require their input to be numerical values. We therefore have to encode our data to make it become useful.

1) Ordinal encoding: Ordinal encoding assigns a real value to each letter. For example, one encoding map could be

$$\text{A} \leftrightarrow 1,\ \text{C} \leftrightarrow 2,\ \text{G} \leftrightarrow 3,\ \text{T} \leftrightarrow 4,\ \text{n} \leftrightarrow 0.$$

Where n is any value that is not "A", "C", "G" or "T". So the sequence "ACCTG" would be encoded as $[0, 2, 2, 3, 1]$.

2) One Hot encoding: One Hot encoding associates each letter with a higher dimensional vector. For example

$$\text{A} \leftrightarrow \begin{bmatrix}1\\0\\0\\0\end{bmatrix}, \text{C} \leftrightarrow \begin{bmatrix}0\\1\\0\\0\end{bmatrix}, \text{G} \leftrightarrow \begin{bmatrix}0\\0\\1\\0\end{bmatrix}, \text{T} \leftrightarrow \begin{bmatrix}0\\0\\0\\1\end{bmatrix}.$$

This would turn a sequence into a matrix, but alternatively we can also take the transpose of the correspondent vectors and concatenate them to get a single vector. Depending on the machine learning algorithm one method might be better suited than the other.

This encoding is widely used in deep learning methods like convolutional neural networks, especially for natural language processing since it does not distinguish between states by making one a higher value, i.e. more important[kC20][Wik23a].

3) k-mer: k-mer is a method of data augmentation that takes in a DNA sequence and returns a set of subsequences of length k. If we had a DNA sequence of "ATGCATGGGT-CAAGFT", 6-mer returns the set "'ATGCAT', 'TG-CATG', 'GCATGG', 'CATGGG', 'ATGGGT', 'TGGGTC', 'GGGTCA', 'GGTCAA', 'GTCAAG', 'TCAAGFT'".

B. Feature extraction

1) Neural networks: We use a PyTorch implementation for a fully connected linear, a convolutional neural network or an autoencoder. For the fully connected and convolutional neural network we evaluate the cosine distance (II-D1) of each data point and use that as a truth value to train the networks on. This approach is a crude method to approximate an autoencoder without having to pay the high time cost of training an autoencoder.

2) NLP model - DNABERT: DNABERT is a NLP based model that uses pre-training on DNA sequences to provide a solution for predicting and understanding various regulatory elements, i.e. specific parts of DNA sequences that play a crucial role[JD21].

3) LSTM: Long short-term memory (LSTM) is a recurrent neural network with an architecture of that allows it to learn and remember information over long sequences. Because of this property we include it in the models we analyze[Con23] - it allows to preserve the information about ordering of the words and in NLP model helps in getting semantic meaning of the words. We implemented Siamese neural networks, which take two sequences and as an output trying to return sequences, which are as similar in the meaning of cosine similarity, as good is pairwise alignment between them.

C. Clustering

1) k-means: We use implementation of the k-means algorithm from the scikit-learn library.

2) DBSCAN: DBSCAN is one of the most used clustering algorithms. In essence, it clusters together points such that all points within a cluster are at most a distance of ϵ from another point of the cluster and so that most points have at least a number of $minPts$ within a distance of ϵ.

This algorithm has numerous advantages, for example not having to specify the number of clusters or being able to find clusters with an arbitrary shape. However, without knowledge of the data, it is difficult to find values for ϵ and $minPts$ such that the algorithm performs well[MX96].

D. Loss functions

Our goal is to be able to extract primers from each cluster. For this to be feasible we need good alignment of the DNA sequences in each cluster. The tool which we were asked to use by the Genorobotics team for analyzing sequence alignment is called Clustal. As we will discuss in Section III, Clustal alone is not enough to have a useful metric to quantify our wanted results. Independently from this, even tough Clustal is a highly efficient algorithm with time complexity of $\mathcal{O}(N \log N)$, where N is the number of sequences, it is still very slow to run taking multiple hours to run once on the full data set of $\sim 90'000$ sequences. We therefore found a proxy metric to still be able to train our models and we will still evaluate the results of each model with Clustal in order to rate them. Because we still need to run Clustal, we sample a set of $50'000$ data points from our original data.

1) Cosine distance: Given two $n-$dimensional vectors A, B, the cosine distance is defined as

$$D_c(A, B) = 1 - \frac{A^\top B}{\|A\|_2 \|B\|_2}$$

note that the cosine distance is not a distance metric but it is still useful because it is based on the cosine similarity which has been shown to outperform other loss functions like MSE, in certain circumstances like the miniBERT model, which is similar to the DNABERT model[HZ20].

2) Pariwise distance: Running Clustal each time we want to check performance of specific set of hyperparemetrs during their tuning is really time consuming. Because of that, we decided to take mean of pairwise distance between points assigned to one cluster as an estimation of clusters quality when choosing number of clusters. These two values are highly correlated with each other (Pearson correlation coefficient between mean pairwise distance and percentage of gaps in sequence at the level of 0.8-0.9 with very low p-value).

E. Self-organizing map (SOM)

The Self-organizing map (SOM) is an unsupervised machine learning method. It is used to produce a low-dimensional representation of a high-dimensional data set, while trying to preserve it's topology. It does this by creating a grid of so called neurons that live in the chosen low-dimensional space and each neuron has an associated weight vector that lives in the same high-dimensional space as the data. Then it iteratively adjusts these weights. At each step, we chose randomly a point from the data set and then search for the neuron with the closest weight vector to that point. The found neuron is called the best matching unit (BMU). The weights of all the neurons are updated based on its distance to the chosen point, its distance to the BMU and a learning rate factor so that the closer the corresponding weight of a neuron is to the BMU, the more its weight gets shifted towards the chosen point[Wik23c][Mor22].

After convergence or a given amount of iterations, we can assign each point from our data to it's closest node representative to get the low-dimensional representation. This representation can then be used with clustering algorithms or, if the number of neurons is equal to the number of clusters we want, we can interpret each neuron as a cluster. This gives us a method that combines the feature extraction and the clustering steps.

III. Evaluation metric for clusters

As mentioned, Clustal is not sufficient to determine if our model produced a good result. Clustal returns an alignment of a cluster from which we can calculate consensus percentage f at each step of the sequence and the gap percentage g in a given window of the cluster. From this we construct the following metric.

Given L_{pri}, the length of the primer, p, the minimum consensus percentage, q, the maximum gap percentage, N_{clu}, the number of clusters and N_{seq}, the total number of sequences, our metric is defined in the following way

$$\widetilde{M}(f_i, g_i) =$$

$$\left[\max_{0\leq m\leq L_{clu}-L_{pri}} \left(\mathbb{1}\{g_i(m) \leq q\} \sum_{n=1}^{L_{pri}} \mathbb{1}\{f_i(m+n) > p\} \right)\right]$$

$$M = \frac{1}{N_{clu}} \sum_{i=1}^{N_{clu}} \frac{N_i}{N_{seq}} \widetilde{M}(f_i, g_i)$$

Where f_i is the function of consensus percentage and $g_i(m)$ is the gap percentage in the window $(m, m+L_{pri})$ the of cluster i, L_{clu} is the length of the aligned cluster and N_i is the number of sequences in cluster i.

This metric is motivated by the inputs of the Genorobotics team. For primer extraction they require that a part of the aligned sequence (length $L_{pri} \approx 45$) has a good enough consensus percentage ($p \approx 0.6\overline{6}$) in most of that part and that it does not surpass a given gap percentage ($q \approx 0.25$). $\widetilde{M}(\cdot, \cdot)$ measures the largest accumulation of points with good properties within a given cluster. The bigger this measure, the higher the probability for the Genorobotics team to be able to extract a primer from this cluster. But we also want to penalize very small clusters with good alignment and very large clusters with bad alignment. We therefore multiply by the relative size of each cluster and take the mean to get an $M-$score that reflects the desired properties.

IV. Results

In the literature there are only a handful of scientific papers that combine DNA barcoding with machine learning and we have been unable to find any that are not for plant identification but for the research of universal primers. This is why we are unable to provide a baseline comparison and quantify the benefits of our methods. However we run k-means on our data provide a baseline for the rest of our results.

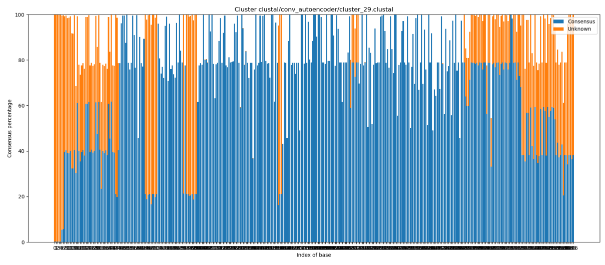

Fig. 1. Example of Alignment with Clustal

A. Illustration of Clusters

Here we provide some visualization of our clusters. In the figure above1 orange bars show gaps in alignment and blue bars suggest successful alignment. The y axis shows the percentage of gaps/alignment within the cluster. The x axis shows the alignment sequence. In the figure above, we have a good alignment in the middle of the sequence and a lot of gaps at the ends. Even though this cluster has reports 15 percent gaps, it is a relatively good cluster for primer extraction as there is a subsection with high alignment.

B. Data preprocessing

For data preprocessing our approach was relatively simple. We needed to encode our DNA sequences to be able to use ML models and clustering algorithms, as mentioned in Section II-A.

As we only require the beginning and the ends of DNA sequences to be aligned, for our testing we took an an arbitrary length of 300 base pairs from the beginning of each sequence. This number ensures that there is enough length in the middle for DNA barcoding purposes and that the sequences are long enough to extract primers from them. We confirmed with our supervisor that this is reasonable approach.

C. DBSCAN

As discussed in Section II-C2, DBSCAN requires knowledge of the data to get good results. After embedding, it is impossible to acquire this knowledge. We used a grid search algorithm to try to get a reasonable amount of clusters but hyper parameters are highly sensitive in our case and we were unable to get good results.

D. SOM

The results for the SOM approach left a lot to desire. It is difficult to say why that is the case but do to time constraints, we were unable to calculate the $M-$score (Clustal takes to long to run).

E. Neural Networks

We started with a simple 2-layer fully connected neural network to have a baseline for our models, as excepted it performs very similar to k-means. Since we are working with sequences, we decided to use CNNs to keep contextual information about the sequences. This provided better results, however as we discuss at the end of this section, we were not satisfied with the distribution of the clusters.

One big concern was that our approach with the cosine distance would not lead to the best results possible. One possible tool to solve this are autoencoders, although they come with their own set of problems, as discussed in II-B1. We used the target decoded sequences to be the same as the original, which enabled us to get embeddings for our sequences that represent the original sequence. We tried using both convolutional and fully connected layers in our autoencoders, as we previously discussed convolutional version performed better as it keeps information about the context.

Upon discussing our task with some Genorobotics members, they suggest us to try DNABERT[JD21]. We tried to use the pre-trained model on 6-mer sequences, which did not provide good results.

As it was not possible to fine tune DNABERT with our data (as we do not have labeled data points), we decided to try the fine tuned model provided in the DNABERT repository. Even though it is not fine tuned for our purpose, there are enough underlying similarities in our task to motivate our use. Our results justify this approach, as fine tuned DNABERT6 was one our best performing models.

The lstm model performance was one of the worst. The clustering done on the results of this type of neural-network gave few very small clusters with high M-score and many big clusters that wasn't feasible to extract primers from them.

You can see our results in the table belowI

TABLE I
K-MEANS CLUSTERING RESULTS (ROUNDED TO 3 SIGNIFICANT DIGITS)

Method	$M-$score
Simple k-means	1.65
Ordinal Linear (22 clusters)	1.13
Ordinal Linear (50 clusters)	0.66
One Hot Linear	2.48
One Hot Convolutional	3.65
Ordinal Linear Autoencoder	1.34
One Hot Convolutional Autoencoder	1.07
k-mer DNABERT6	0.57
k-mer Fine Tuned DNABERT6	1.04
k-mer LSTM	0.16

As you can see in the table aboveI, k-means, one hot linear and one hot convolutional are the ones with the highest scores. However, these three models each have a cluster with 60 percent of the sequences. Even though we tried to penalize very big clusters with bad alignment in our metric, in our opinion, it does not penalize them enough. This is why we chose our convolutional autoencoder as our best model as it still achieves a good score and has more even clusters.

To give our results more robustness, we ran validation on the rest of our data with the one hot convolutional autoencoder and we get very similar results to our training set, our $M-$score for validation was 1.14. We have not done this with the rest of our models because of our previously mentioned computational cost of running Clustal. This shows that our method might also be applicable with different data sets.

As a final test, we were asked by our supervisor to check if our methods cluster the same plants together. On our best performing model, we looked at the number of plant families in each cluster, we have a total of 14425 plant families in our data set and the the maximum number of families in a cluster in autoencoder is about 3000. Meaning that they are pretty evenly distributed, which is what they were hoping for.

V. ETHICAL RISKS

One can easily imagine a world where these kinds of methods are used to analyze human DNA. For example it could be misused in forensics, there are potential risks in genetic privacy[WB23] and discrimination based on genetic traits[Ott97].

Strictly speaking, the ethical issues arise from the PCR method and not directly from our machine learning model. However since we are helping the PCR method becoming more efficient, although indirectly, we contribute to these issues.

Focusing on the discrimination based on genetic traits, currently it is not economically viable for medical insurance companies to sequence the DNA of the insurance holders. If however the cost of sequencing the parts of the DNA that are interesting to the company, i.e. genes that have a negative impact on medical health, decrease enough, it could lead to the companies charging people with these genes more for the insurance or not give them insurance at all.

Therefore, the risk of our method is that it might become possible to find a primer or a group of primers that allows sequencing of multiple genes at the same time so that the costs are reduced enough.

If this were indeed to occur, which is likely if the legislator does not intervene[AS21], the consequences could be severe for the people affected. There are nevertheless encouraging trends in that worldwide already 47 countries, including Switzerland, 23 other European countries and the US, have adopted legislation to resolve the use of predictive genetic information by insurers[YH10].

We were not able to take this risk into account because human DNA and plant DNA are essentially indistinguishable without advanced techniques. Even if we were able to implement a a way to block the usage on human DNA, it is still possible for someone to do their own implementation without this restriction.

VI. SUMMARY

We tried different approaches to clustering for primer extraction, most of which did not give satisfactory results. The best results were obtained using the "One Hot Convolutional Autoencoder" model. However, as this is a novel approach to primer design, there is still a lot of room for improvement and exploration of our results, as well as trying to fine tune the results with a better and a smaller number of clusters.. For example, simply using deeper neural networks might lead to improved results. Additionally, for future work we would like to explore the nature of our clusters, families of plants, genus, geographic region, etc.. If the clusters we get correspond approximately to geographic regions, they could be very useful for Genorobotics' field trips.

Even though we don't have perfect results, our discussion with our supervisor confirmed that our results could be good enough that our methods could used in real world applications, as any reduction in the number of primers required for an expedition is welcome.

References

[AS21] Sonia M. Suter Anya E.R. Prince Wendy R. Uhlmann and Aaron M. Scherer. *Genetic testing and insurance implications: Surveying the US general population about discrimination concerns and knowledge of the Genetic Information Nondiscrimination Act (GINA)*. Online. Accessed: December 21, 2023. 2021. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9165621/.

[Con23] PyTorch Contributors. *LSTM*. Online. Accessed: December 21, 2023. 2023. URL: https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html.

[HZ20] Yi-Chin Huang and Xiaomiao Zhang. *Enhancing minBERT for Sentence Similarity with Cosine Similarity and Contrastive Learning*. Department of Computer Science Stanford University. Accessed: December 21, 2023. 2020. URL: https://web.stanford.edu/class/cs224n/final-reports/final-report-169591022.pdf.

[JD21] Han Liu anrong Ji Zhihan Zhou and Ramana V Davuluri. *DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome*. Bioinformatics, Volume 37, Issue 15, August 2021, Pages 2112–2120. Accessed: December 21, 2023. 2021. URL: https://doi.org/10.1093/bioinformatics/btab083.

[kC20] Akash kumar and Nagesh Singh Chauhan. *DNA Sequencing with Machine Learning*. Online. Accessed: December 21, 2023. 2020. URL: https://www.kaggle.com/code/singhakash/dna-sequencing-with-machine-learning.

[KE07] W. John Kress and David L. Erickson. *A Two-Locus Global DNA Barcode for Land Plants: The Coding rbcL Gene Complements the Non-Coding trnH-psbA Spacer Region*. PLoS ONE 2(6): e508. doi:10.1371/journal.pone.0000508. Accessed: December 21, 2023. 2007. URL: file:///C:/Users/Younis/Downloads/Telegram%20Desktop/Kress_et_Erickson_2007_A_Two_Locus_Global_DNA_Barcode_for_Land_Plants.pdf.

[Mor22] Ken Moriwaki. *Understanding Self-Organising Map Neural Network with Python Code*. Online. Accessed: December 21, 2023. 2022. URL: https://towardsdatascience.com/understanding-self-organising-map-neural-network-with-python-code-7a77f501e985.

[MX96] Jiirg Sander Martin Ester Hans-Peter Kriegel and Xiaowei Xu. *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databaseswith Noise*. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. Accessed: December 21, 2023. 1996. URL: https://cdn.aaai.org/KDD/1996/KDD96-037.pdf.

[Ott97] Carla Otto. *Ethical Issues of Genetic Screening*. Accessed: December 21, 2023. 1997. URL: https://www.ndsu.edu/pubweb/~mcclean/plsc431/students/otto.html.

[QS17] Josh Quick and Andy Smith. *primer panels for multiplex PCR*. PLoS ONE 2(6): e508. doi:10.1371/journal.pone.0000508. Accessed: December 21, 2023. 2017. URL: https://primalscheme.com/.

[WB23] Joby Warrick and Cate Brown. *China's quest for human genetic data spurs fears of a DNA arms race*. "The Washington Post". [Online; accessed 07-December-2023]. 2023. URL: https://www.washingtonpost.com/world/interactive/2023/china-dna-sequencing-bgi-covid/.

[Wik23a] Wikipedia. *One-hot*. [Online; accessed 17-October-2023]. 2023. URL: https://en.wikipedia.org/wiki/One-hot.

[Wik23b] Wikipedia. *Polymerase Chain Reaction - Applications*. Accessed: December 21, 2023. 2023. URL: https://en.wikipedia.org/wiki/Polymerase_chain_reaction#Applications.

[Wik23c] Wikipedia. *Self-organizing map*. [Online; accessed 17-October-2023]. 2023. URL: https://en.wikipedia.org/wiki/Self-organizing_map.

[YH10] Maria Braker Yann Joly and Michael Le Huynh. *Genetic discrimination in private insurance: global perspectives*. New genetics and society. Accessed: December 21, 2023. 2010. URL: https://www.ndsu.edu/pubweb/~mcclean/plsc431/students/otto.html.