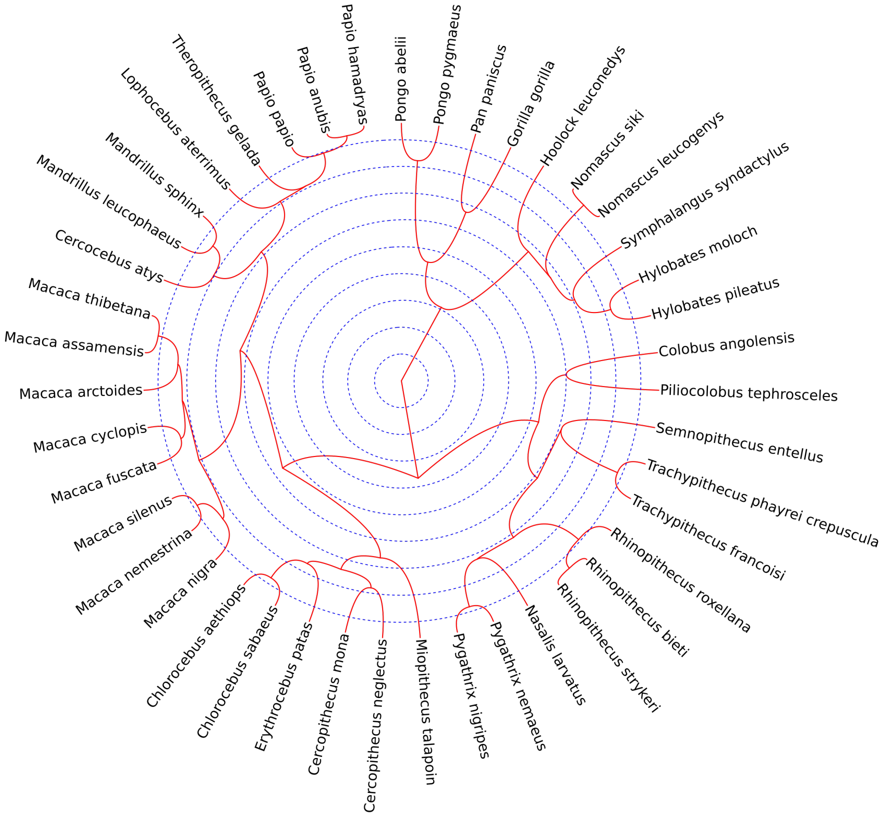


Predicting evolutionary distances from variable length Markov chains with deep regression

Master's thesis in Computer science and engineering

Filip Helmroth
Erik Söderpalm

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2023

Master's thesis 2023

Predicting evolutionary distances from variable length Markov chains with deep regression

Filip Helmroth
Erik Söderpalm

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

Predicting evolutionary distances from variable length Markov chains with deep regression
Filip Helmroth & Erik Söderpalm

Supervisor: Alexander Schliep, Department of Computer Science and Engineering
Examiner: Marina Axelson-Fisk, Department of Mathematical Sciences

Master's Thesis 2023
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A phylogenetic tree of 43 different species from the parvorder Catarrhini, a dataset used in this thesis.

Typeset in LaTeX
Gothenburg, Sweden 2023

Predicting evolutionary distances from variable length Markov chains with deep regression
Filip Helmroth & Erik Söderpalm
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

In light of the accelerated surge of sequence data due to next-generation sequencing technologies, traditional alignment-based approaches for genome comparison are being outpaced, leading to a rising interest in more efficient alignment-free comparison methods. One such method is the alignment-free method based on Variable Length Markov Chains (VLMCs).

In this thesis, we explore the application of VLMCs as genomic signatures to estimate evolutionary distances. We employ deep regression models and alignment-free VLMC distances which are computed through a recently developed distance measure (d_v^*) for VLMCs.

Genomic data from over 300 various species are downloaded from the National Center for Biotechnology Information database and are used to train multiple deep regression models. The thesis is structured in two complementary parts. First, we develop and evaluate models for estimating evolutionary distances through VLMC distances on synthetic mutations. Second, we develop models for estimating divergence times of various species using VLMC distances and data derived from TimeTree, a public knowledge base derived from thousands of published studies.

The results show that regression models built on carefully selected features outperform linear regressor benchmarks in predicting evolutionary distances in both parts of the project and across all evaluation metrics. The thesis effectively demonstrates the promising results of VLMCs and d_v^* in deep regression models for predicting evolutionary distances and highlights potential areas for future research to improve model accuracy.

Keywords: Computer science, Bioinformatics, Master's thesis, Markov chains, Variable length Markov chains, Deep regression, Evolutionary distance, Alignment-free sequence analysis, Genomic signatures, DNA

Acknowledgements

We wish to express our gratitude to our supervisor Alexander for his guidance throughout this thesis and for his much-appreciated genuine enthusiasm. We would also like to thank Joel, who has been a great sounding board and source of knowledge on the subject of Variable Length Markov Chains. Lastly, we would like to thank Marina, our examiner.

Filip Helmroth & Erik Söderpalm, Gothenburg, June 2023

Contents

List of Figures

List of Tables

1
Introduction

Genomic sequence analysis is a powerful tool for understanding the evolutionary relationships among various organisms and viruses. Measuring evolutionary distances based on genomic differences provides insights into how species have evolved and diversified over time. However, evolutionary distances are often derived in a manual process usually from comparing evolutionary changes in a small number of gene sequences. This requires expertise since a selection needs to be made on which genes to consider.

However, despite these traditional methods' success in measuring evolutionary distances, they can be very time-consuming. Consequently, new strategies have been developed using representations of an organism's genome, called Genomic Signatures. In bioinformatics, one common type of genomic signature is k-mers. A k-mer is a contiguous substring of length k. They can be thought of as a small "word" that represents a subsequence within a genome.

A group, led by Peter Norberg, has developed Genomic Signatures based on Variable Length Markov Chains (VLMCs) for classifying and comparing genomes [12, 13, 14]. In essence, VLMCs contain k-mers of varying k. Their main advantage is their ability to model the complexities of genomic sequences and their increased statistical robustness.

The distances between two VLMCs correlate to a certain degree with their respective genomes' evolutionary distance, however, they can deviate. Tang *et al.* addressed this issue for k-mers by proposing a method that uses deep regression networks to correct these deviations [15]. Their method showed good performance in estimating the evolutionary distance from genomic sequences. Consequently, this suggests that a similar problem could be investigated using VLMC distances.

1.1 Purpose

This thesis builds upon a current project of Gustafsson *et al.* and previous work by Tang *et al.* [15, 16]. It aims to demonstrate the potential of VLMCs and VLMC distances in predicting different definitions of evolutionary distance using deep regression. In more detail, the project is divided into two parts. The first part explores VLMCs as an alignment-free approach with evolutionary distance defined as the Levenshtein distance, similar to Tang *et al.*'s work with k-mers [15]. The second part explores if VLMCs and VLMC distances can be used to predict the number of years since two species diverged based on their genomes, i.e. their divergence time.

1.2 Limitations

The project will only consider genomic signatures based on VLMCs. Additionally, the project will only consider one distance function between VLMCs, namely d_v^*, which is a state-of-the-art algorithm that can process large datasets without requiring impractical amounts of memory [16].

2

Background

This section will cover both the biological and technical details needed to understand the fundamentals of the work in this thesis.

2.1 Genetic background

Understanding the foundational concepts of DNA, biological classification and genomic signatures is crucial to grasp the background. Below, we introduce each idea.

2.1.1 DNA

DNA, or deoxyribonucleic acid, is a double-stranded molecule that serves as the genetic blueprint for all living organisms. DNA is found in almost all living cells and contains the instructions necessary to function and reproduce. The molecule consists of four bases: adenine (A), cytosine (C), guanine (G), and thymine (T). These bases are arranged in a specific sequence along the DNA strand as shown in Figure 2.1, which determines the genetic code of the organism. The double-stranded form of DNA is complementary, meaning that A always pairs with T and G always pairs with C. Hence, both strands of these base pairs (BPs) contain the same information.

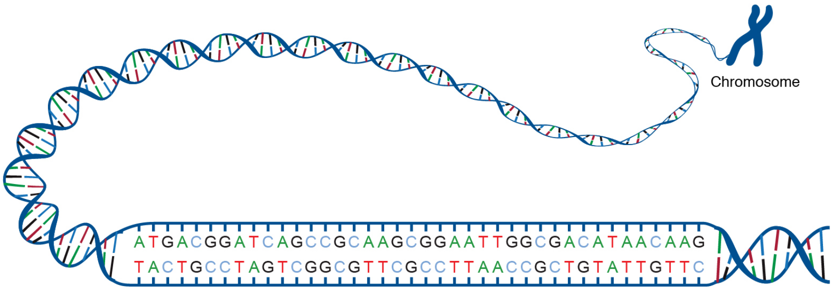

Chromosome
ATGACGGATCAGCCGCAAGCGGAATTGGCGACATAACAAG
TACTGCCTAGTCGGCGTTCGCCTTAACCGCTGTATTGTTC

Figure 2.1: A depiction of DNA, showing how adenine (A) and thymine (T), and guanine (G) and cytosine (C) pair up, respectively. (Source: [1]).

2.1.2 Biological classification

Biological classification is a systematic approach to organizing and categorizing living organisms based on shared characteristics and evolutionary relationships. Phylogenetic trees are branching diagrams that illustrate these relationships. Organisms are arranged in a taxonomic hierarchy, which refers to naming and classifying organisms into hierarchical categories or taxonomic ranks, such as domain, kingdom, phylum, class, order, family, genus, and species.

For example, the moose (*Alces alces*) is classified within the class Mammalia (mammals), order Artiodactyla (even-toed ungulates), and family Cervidae (deer and related species). Its genus is Alces, and the species name is alces, hence leading to the name *Alces alces*.

Figure 2.2 shows an example of a phylogenetic tree of 43 different species from the parvorder Catarrhini, or old world monkeys. Here, for example, *Macaca Silenus* is estimated to have diverged from *Gorilla Gorilla* about 29 million years ago (MYA).

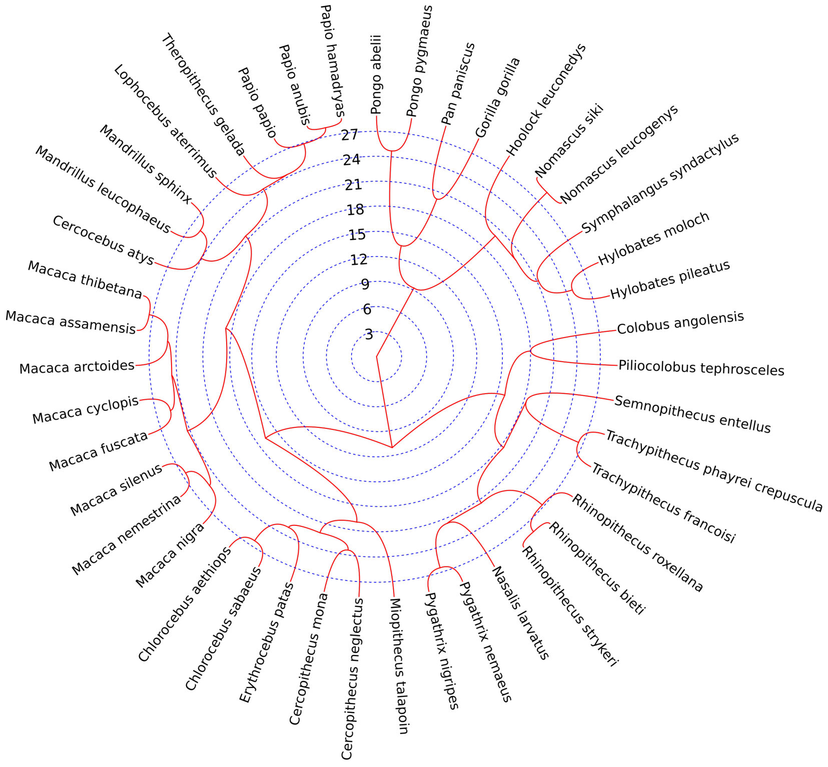


Figure 2.2: A phylogenetic tree of 43 different species from the parvorder Catarrhini.

2.1.3 Pathogens

Pathogens are microorganisms that cause diseases by infecting host organisms. It is a collective word for a diverse group of bacteria, viruses, and fungi. Despite having considerably shorter genomes, most pathogens contain DNA similar to humans. For instance, closely related pathogens, such as different forms of *Escherichia coli* (E. coli), can be identified by examining the similarities in their genetic composition. Accurate taxonomic classification of pathogens is essential for understanding their evolutionary relationships which informs targeted strategies for disease control [17].

2.1.4 Genomic Signatures

Genomic signatures are recurring patterns or features in the genome that can be used to differentiate between organisms, classify genomes, and study evolutionary relationships. These signatures can be generated using methods such as k-mer counting, Markov models, and other statistical models, which are discussed below. The purpose of computing genomic signatures is to enhance the efficiency of genome-related algorithms, such as classification, by identifying key features within the genome.

Holmudden has demonstrated that genomic signatures, based on variable length Markov chains, can accurately differentiate between various viral species and capture critical features of virus DNA [18].

GC content is another genomic signature that refers to the proportion of G and C bases in a genome. It is commonly used as a genomic signature as GC-rich regions have other characteristics compared to AT-rich regions, which can aid in differentiating species [19]. As a result, by incorporating GC content analysis into the analysis, the accuracy of genome-related algorithms can increase. The GC content can be calculated as a percentage of the total number of base pairs in the genome.

2.1.5 Taxonomy database

Sequencing data from varying organisms will primarily be gathered from the National Center for Biotechnology Information (NCBI) taxonomy database [20]. The database consists of the names of organisms, the length of the sequence, and their corresponding GenBank assembly accession. The GenBank assembly accession is a unique identifier for each genome, which serves as a reference for locating and retrieving the corresponding genomic data.

2.2 Markov chains

As mentioned above, Markov models are often used in sequence analysis. A Markov chain is a stochastic model that follows the Markov property, which states that the next state of the system depends on a set amount of present and previous states [21]. The model is a discrete-state random process, defined over a sample space Σ with a set of discrete states $\Sigma = \{X_1, X_2, \ldots, X_n\}$. In the field of genomics, Markov chains can be used to model the probabilistic structure of DNA sequences, and they are often used to represent genomic signatures.

2.2.1 Fixed order Markov chains

A *first-order* Markov chain depends solely on the present state and not the history. The transition property is defined as

$$P(X_n = x_n | X_{n-1} = x_{n-1}) \tag{2.1}$$

where the Markov chain only depend on the states x_n and x_{n-a}. The transition probability $P(x_n|x_{n-1})$ states the probability of traversing from x_{n-1} to x_n. Consider an example of a first-order Markov chain in the context of genomics. The sample space $\Sigma = \{A, T, C, G\}$ represents the four bases in DNA. In Figure 2.3, a graphical representation of the sequence *TGCAAC* is shown.

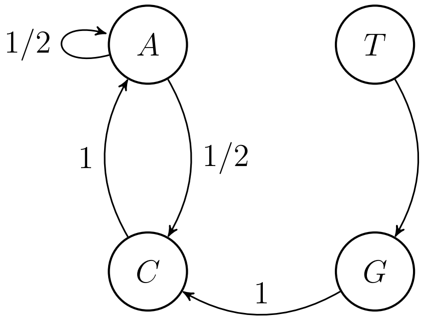


Figure 2.3: A first-order Markov chain representing the sequence *TGCAAC*, where only one character is considered as the previous state.

In the example, only one character is considered when evaluating the next state. The transition probability $P(x_n|x_{n-1})$ of X_n can be expressed in a transition matrix. For a first-order Markov chain with $N = |\Sigma|$ states, the transition matrix have the size $N \times N$.

A *higher-order* Markov chain is an extension of the first-order Markov chain, where the probability of transitioning between states depends not only on the current state but also on a predetermined number of preceding states called *context*. In a Markov chain of order k, the future state relies on the present state and the previous $k - 1$ states, mathematically defined as

$$P(X_n = x_n | X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}, \ldots, X_{n-k} = x_{n-k}) \tag{2.2}$$

Due to the long history of states, the model can capture more complex dependencies in the sequences [22]. For a higher-order Markov chain with $N = |\Sigma|$ states, the size of the transition matrix will increase exponentially in the size of N or in the order of N^{k+1}.

Considering the same sequence *TGCAAC* as in Figure 2.3, but for a *third* order Markov chain, the transition probability from GCA to A is 1 and 0 to T, G, and C.

In bioinformatics, k-mers are often used in sequence analysis, since they are powerful tools for identifying similarities between biological sequences. Essentially, k-mers act as representations or states within fixed-order Markov chains of order $k-1$. For example, the 4-mer $TGCA$, the 1-mer T, or the 6-mer $TGCAAC$ each symbolize states in a Markov chain. Furthermore, the transitions between these states express the progression from one k-mer to another in the sequence, capturing dependencies in genomic sequences.

In addition, k-mer counting (the frequency of each unique k-mer in the sequence) provides information to compute transition probabilities between states in the Markov chain. A high frequency of a certain k-mer indicates a high transition probability between its represented states in the Markov chain. Therefore, the k-mer count plays an important role when constructing a Markov chain of the sequence.

2.2.2 Variable length Markov chains

Similarly to fixed-order Markov chains, Variable length Markov chains (VLMCs) are probabilistic models that capture the statistical properties of sequences [12]. However, VLMCs allow the length of the context to vary between different states rather than being fixed. Hence, the order k varies between different states and the order of a VLMC is defined as the length of the longest context.

The order k of each state depends on the sequence's recent history and is determined using statistical information derived from the sequence. This statistical information helps determine whether to include or exclude states based on a pruning threshold, such as state frequency. For example, a VLMC where $\Sigma = \{A, C, G, T\}$ could include only the transition probabilities $P(\Sigma|ATCC)$ and $P(\Sigma|TCC)$ of context length 4 and 3. This is due to the fact that other states with the specific context length do not meet certain thresholds for the number of occurrences.

A VLMC is an efficient method for modeling organisms with long and important sequences [14]. The model can extract important sequences and prune less important sequences, which can be critical for distinguishing between species. The number of stored states in a higher-order Markov would be N^{k+1}, but where a VLMC would need to store significantly fewer states due to pruning.

Schulz *et al.* presented an algorithm for constructing VLMCs [23]. In summary, the construction of the VLMC for a given sequence S involves using lazy suffix trees. The algorithm takes three parameters as input:

1. The minimum frequency allowed for a sequence to be included
2. The maximum length allowed for a context to be included
3. A threshold value, which prunes the number of sequences in the tree

Consider an example of a VLMC with sample space $\Sigma = \{A, T, C, G\}$. Minimum frequency allowed is 2, maximum length allowed is 3, and 0 threshold value. Below, a graphical representation of an example sequence is shown.

$$G\ A\ T\ C\ C\ A\ T\ C\ C\ A$$

The words with a minimum frequency of 2 are *{A, T, C, AT, TC, CC, CA, ATC, TCC, CCA, ATCC, TCCA, ATCCA}*. However, *{ATCC, TCCA, ATCCA}* exceeds the length allowed and hence gets deleted. Including the root node ϵ, the VLMC will have 11 states.

Further, Gustafsson's work includes a definition for VLMC size [24], which is the sum of the length of all k-mers of the VLMC, Σ_{VLMC}. More formally,

$$\sum_{w \in \Sigma_{VLMC}} len(w).$$

2.3 Distance functions

Distance functions represent an essential tool for comparing and measuring the degree of similarity or dissimilarity between genetic sequences. We present two distance functions that will be used throughout the thesis, namely evolutionary distance and VLMC distance.

2.3.1 Evolutionary distance

Evolutionary distance is a measure of the amount of genetic divergence between two sequences, often expressed as the number of differences in nucleotide or amino acid sequences. It is influenced by both the rate of genetic change and the time since the organisms diverged from a common ancestor.

Evolutionary distance is commonly measured by comparing the sequences through alignment-based methods and counting the number of differences between them, generally known as Levenshtein Distance or edit distance [25]. A generalized Levenshtein distance approach involves assigning varying costs to operations such as substitutions, insertions, and deletions.

Another definition of evolutionary distance is divergence time. Divergence time refers to the amount of time that has elapsed since two species separated from a common ancestor. Using the toolkit TimeTree, which is based on more than 3 000 studies, the resource provides access to divergence times of over 97 000 species derived from molecular sequence data [11]. For example in the domain eukaryotes, the pomegranate *punica granatum* and the green sea turtle *chelonia mydas* are distanced 1530 MYA (million years ago). Meanwhile, the pomegranate *punica granatum* and the apple *malus sylvestris* are distanced 108 MYA.

These two definitions can seem quite different as one considers mutations in the genome, while the other considers time. However, they can be related through the concept of a *molecular clock* [26]. The molecular clock is a technique that estimates the time of species divergence, based on the rate of mutations. As mutations accumulate in the genomes over time, the molecular clock can be used to establish a relationship between genetic variation and time.

2.3.2 VLMC distance

The stochastic nature of VLMCs presents a challenge when it comes to measuring the distance, as two VLMCs that have different structures may still be able to model a similar underlying sequence. This is because the probability distributions of the VLMCs can change depending on the specific sequence being modeled, making it difficult to directly compare the VLMCs themselves.

To our knowledge, no optimal distance function exists for VLMCs due to the varying advantages and disadvantages of each function. Prior work on VLMCs has evaluated multiple distance functions including *Kullback-Leibler distance*, *frobenius norm* and *PST-matching* [27].

In a subsequent paper, Gustafsson *et al.* presented a VLMC distance function d_v^*, which uses a modified version of the d_2 measure initially proposed by Torney *et al.* [16, 28]. In short, the distance function is extending an alignment-free dissimilarity measure for k-mers to the next-symbol probabilities in VLMCs.

The following equations present in detail how the distance function d_v^* operates:

$$d_v^*(\lambda_i, \lambda_j) = \frac{2\arccos\left(D_v^*(\lambda_i, \lambda_j)\right)}{\pi}, \tag{2.3}$$

$$D_v^*(\lambda_i, \lambda_j) = \frac{\sum_{w \in C} \sum_{\sigma \in \Sigma} \hat{p}^M(\sigma|w, \lambda_i)\hat{p}^M(\sigma|w, \lambda_j)}{\sqrt{F_v^*(\lambda_i)}\sqrt{F_v^*(\lambda_j)}}, \tag{2.4}$$

$$F_(^*\lambda_i) = \sum_{w \in C} \sum_{\sigma \in \Sigma} \hat{p}^M(\sigma|w, \lambda_i), \text{ and} \tag{2.5}$$

$$\hat{p}^M(\sigma|w, \lambda_i) = \frac{\hat{p}(\sigma|w, \lambda_i)}{\sqrt{\hat{p}(\sigma|w[|w| - M : |w|], \lambda_i)}}. \tag{2.6}$$

Here, λ_i and λ_j are the two VLMCs being compared. Σ is in this algorithm a set of $\{A, C, G, T\}$. C is the set of shared contexts w between the two VLMCs. $\hat{p}(\sigma|w, \lambda)$ is the estimated probability of observing symbol σ in each VLMC given the context w. M is the background-order that normalizes probabilities for next-character probabilities $\hat{p}$ and $\hat{p}^M(\sigma|w, \lambda)$ is the adjusted transition probabilities of observing symbol σ in a VLMC given the context w and the Markov order M. Finally, $w[|w| - M : |w|]$ refers to a substring of length M obtained by taking the last M characters from w.

2.4 Regression models

Regression models provide a robust framework to relate the distances. A regression model is a type of statistical model used to explore the relationship between one or more independent variables and a dependent variable, or target variable. The aim of regression analysis is to develop a mathematical function from the relationship between the variables which can be used to predict the value of the target variable based on the values of the independent variables.

In this section, we will discuss a few different models for regression problems as the one explored in this thesis, as well as how to measure and evaluate their performance.

2.4.1 Linear regression

The most common type of regression model is the linear regression model, which assumes a linear relationship between the independent variables and the target variable. The goal of linear regression is to find the best-fitting straight line through the data that minimizes the distance between the observed values and the predicted values.

In *simple* linear regression, there is only one independent variable and one dependent variable. The linear equation is written as

$$y = \beta_0 + \beta_1 x + \epsilon,$$

where y is the dependent variable, x is the independent variable, β_0 is the intercept of the regression line, β_1 is the slope of the regression line, and ϵ is the error term, representing the difference between the observed and predicted values.

Multiple linear regression is a generalization of simple linear regression, where there are two or more independent variables. As such, the linear equation is similar to simple linear regression and is written as

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n + \epsilon,$$

where $x_1, x_2, ..., x_n$ are the independent variables and $\beta_1, \beta_2, ..., \beta_n$ are the slopes of the regression line for each independent variable. Multiple linear regression can be used to model more complex relationships between the independent and dependent variables, as more features are taken into consideration.

2.4.2 Deep regression networks

To model non-linear relationships, one can instead use deep regression. A deep regression network is a type of neural network, designed to predict continuous output values, also known as regression targets. These networks are called *deep* because they are composed of multiple layers of neurons. The layers are connected in a way that allows information to flow forward through the network, from the input layer to the output layer.

In a deep regression network, the input layer receives input values, and these values are processed through one or more hidden layers before being outputted as a prediction. The hidden layers can consist of various types of neurons, including fully connected, convolutional, or recurrent neurons.

The goal of a deep regression network is to learn a non-linear function that maps input features to output values. This function can be used to make predictions on new input data that the network has not seen before. The training of a deep regression network involves minimizing the difference between the predicted output values and the actual target values through an optimization process such as gradient descent.

Deep regression networks have been used in a wide range of applications, including image and speech recognition, natural language processing, and predictive modeling. Their ability to learn complex non-linear relationships between input and output variables makes them a powerful tool for many prediction tasks.

2.4.3 Measuring performance

In regression analysis, the predicted value is an estimate of the true value (also called ground truth). Hence, an approximation is required for evaluating performance. The mean squared error (MSE) measures the average squared difference between the predicted value and the true value. MSE is defined by

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \tag{2.7}$$

Here, n is the number of samples, y_i is the true value, and $\hat{y}_i$ is the predicted value. A lower MSE indicates a better performance. Further, root mean squared error (RMSE) is the square root of the MSE and provides a more interpretable measure of the error, expressed in the units of the target variable. Due to its sensitivity to outliers, the RMSE may be disproportionately influenced by substantial errors. RMSE is defined by

$$RMSE = \sqrt{MSE} = \sqrt{\frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{n}}. \tag{2.8}$$

Another measure is the coefficient of determination (R^2), which is a common goodness-of-fit measure for regression models and it measures the proportion of variability in y that is described by the variation in x [29]. R^2 is defined by

$$R^2 = 1 - \frac{SSE}{SS_{tot}}. \tag{2.9}$$

Here, $SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ is the sum of squared errors and $SS_{tot} = \sum_{i=1}^{n}(y_i - \bar{y})^2$ is the total sum of squares. $\hat{y}_i$ is the predicted value, y_i is the actual value, and $\bar{y}$ is the mean of the actual values for y. The R^2 score ranges from 0 to 1, where a

higher value indicates a better fit. If the R^2 score is negative, it means that the fitted regression line is worse than the average of the data.
Lastly, Spearman's rank correlation coefficient (Spearman's ρ) is a non-parametric measure of the strength and direction of the monotonic relationship between two variables. It is a useful metric for measuring the performance of a deep regression network when the relationship between the predicted and true output values is non-linear or the distribution of the data is not normal. To use Spearman's ρ for a network, one first computes the predicted output values. Then, the predicted values and the true values are ranked separately, and the correlation coefficient is computed between the ranks. Spearman's ρ is computed as

$$\rho_{R(X),R(Y)} = \frac{\text{cov}(R(X), R(Y))}{\sigma_{R(X)}\sigma_{R(Y)}}, \tag{2.10}$$

where ρ denotes the usual Pearson correlation coefficient but is applied to the rank variables.

2.4.4 Evaluating performance

Generally, in machine learning, one divides data into three sets: a training set, a validation set, and a test set. The purpose of this is to help evaluate the performance of a model and to prevent overfitting.

1. **Training set**: This set is used to train the model. The model is exposed to the input data and the corresponding ground truths and learns to map the input to the output. The training set is used to adjust the weights of the model and to minimize the error between the predicted values and the ground truths.

2. **Validation set**: This set is separated from the training set. It is used to tune the hyperparameters of the model, such as learning rate, the number of hidden layers, or the number of neurons per layer. The validation set provides an estimate of the performance of the model on unseen data and helps prevent overfitting, which is a result of a too complex model that has learned to fit the training data too close to the ground truths, resulting in poor performance on new, unseen data.

3. **Test set**: This set is used to evaluate the performance of the model on unseen data after the model has been trained and validated. The test set provides an unbiased estimate of the performance of the model, and it is important to use a separate test set that is not used for training or validation to avoid overfitting and to ensure that the model can generalize to new data.

Additionally, depending on the total amount of data, the sets are often divided in ratios such as 60:20:20, 70:15:15, or 80:10:10. The validation set size will depend on the number of hyperparameters used in the model, for example, a model with more hyperparameters could need a bigger set for validation. The split is generally done randomly, this is to ensure that all three sets are representative of the overall dataset and that the model is not biased towards a particular subset of the data.

To further ensure that the training and validation sets are representative of the overall dataset, one can use K-fold cross-validation (KFCV) [30]. KFCV divides the dataset into K equally sized and non-overlapping subsets (folds). One subset is used for validation and $K-1$ subsets are used for training the model. The training is iterated K times, with each fold being used for validation once. Figure 2.4 shows the basic idea of cross-validation. The overall performance is evaluated by taking the average of the measures from each iteration, which gives a more accurate estimate of model prediction performance.

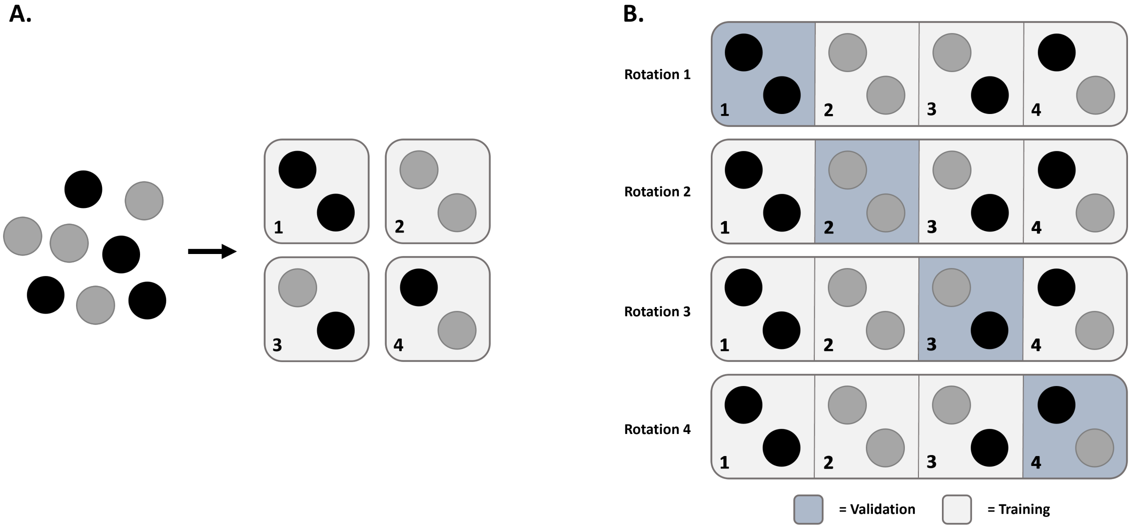


Figure 2.4: The basic idea of cross-validation. **A.** Unstructured data samples (circles) are divided into subsets (folds). **B.** The training and validation is iterated K times, in varying constellations. Note: The colors of the circles are only there to distinguish the different folds.

3

Methods

In this chapter, we present the data and methodology used to conduct the thesis. The chapter is organized into four distinct sections, which include our experimental setup, data collection, features considered for the models, how the models were trained, and lastly how we measured the performance of our models.

3.1 Experimental setup

The code has been developed using Python and is based on Gustafsson's *dvstar* repository [24]. Unless otherwise specified, the *scikit-learn* library (Pedregosa *et al.*, 2011) is used for constructing and implementing various machine learning models.

The project was divided into two parts. The first part involved constructing a regression model to predict evolutionary distance using VLMCs of synthetic mutations and evolutionary distance based on the Levenshtein distance. The other part involved constructing a regression model to estimate the divergence time between species (i.e. evolutionary distance), using VLMCs based on the genomes and evolutionary distances derived from TimeTree.

3.2 Data collection

In this section, we describe the process of our data collection, the computation of evolutionary distances and the calculation of VLMC distances.

3.2.1 Genomic datasets

For the first part with synthetic mutations, we used two pathogens, namely the bacteria E. coli and the virus SARS-CoV-2. For the second part with divergence times from TimeTree, we used three datasets of genomes from various eukaryotes and prokaryotes; one diverse set of 149 prokaryotes, one diverse set of 135 eukaryotes, and one more closely related dataset of 43 eukaryotes from the parvorder Catarrhini. The 149 prokaryotes consisted of 134 bacteria and 15 archaea. The 135 eukaryotes consisted of 44 Animalia, 42 Plantae, and 49 Fungi species. Some examples from the species used in each dataset are shown in Figure 3.1 and 3.2. A complete list of the genomes in each dataset is in Appendix A, B, and C.

Figure 3.1: Examples of species from the broad dataset of 135 eukaryotes, consisting of e.g. a fungi, a cacao tree, a rock grayling, and a red deer (Source: [2, 3, 4, 5]).

Figure 3.2: Examples of species from the dataset of 43 Catarrhini's, consisting of e.g. the Pileated gibbon, the Guinea baboon, the Proboscis monkey, and the Sumatran orangutan. (Source: [6, 7, 8, 9]).

The genomes were downloaded from the National Center for Biotechnology Information (NCBI) database. Each genome sequence was stored as a FASTA-file, which is a widely used standard in the field of bioinformatics and applicable for VLMC construction [31]. Subsequently, using Gustafsson's code, the sequences were translated to VLMCs, and the pairwise distances of the VLMCs were computed.

Below, Table 3.1 shows a selection of organisms in the dataset.

Organism	**Domain**	**Scientific name**	**Genome size (10^6 BPs)**	**GC ratio (%)**	**GenBank ID**
Cacao Tree	Eukaryote	Theobroma cacao	325	31.7	GCF_000208745.1
Central Asian red deer	Eukaryote	Cervus hanglu	2600	40.4	GCA_010411085.1
Guinea baboon	Eukaryote	Papio papio	2908	39.8	GCA_028645565.1
⋮	⋮	⋮	⋮	⋮	⋮
SARS-CoV-2	Virus	Severe acute respirat[...]	0.03	37.4	GCA_009858895.3
⋮	⋮	⋮	⋮	⋮	⋮
E. Coli	Prokaryote	Escherichia coli	4.64	50.0	GCA_000005845.2
Streptococcus	Prokaryote	Streptomyces albus	7.89	71.7	GCF_030011775.1

Table 3.1: Summary of selected organisms and their genomic information.

3.2.2 Evolutionary distance using synthetic mutations

The data for mutations was synthetically created by modifying a genome and replacing characters randomly. We used a tweaked version of the Levenshtein distance [25], where we only considered substitutions as a move in distance. For example, Figure 3.3 shows two examples of the evolutionary distance (in terms of Levenshtein distance) and its corresponding ratio of the sequence $ACGGA$.

A potential limitation of this approach is that the model is exclusively trained on sequences of equal length to the original. However, this constraint avoids the complexities associated with comparing and aligning sequences of varying lengths, which

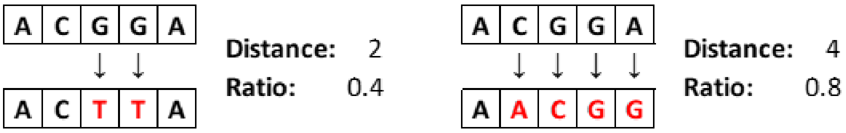


Figure 3.3: Mutations were synthetically created on a per-character basis. Each modification is counted as a step in evolutionary distance, i.e., one step in Levenshtein distance.

could otherwise introduce ambiguity to the analysis.

Gustafsson's previous work has shown that evolutionary changes exceeding roughly 30% of the original tend to be too noisy with VLMC distances to use [10]. Thus, we limited our synthetic sequences to only have up to 22.5% deviation from the original sequence. Additionally, for genomes of greater size, we limited the divergence to even lower percentages. Both because of the time complexity of creating the data and also to simulate more realistic mutations, where the genome does not change as dramatically.

3.2.3 Evolutionary distance using TimeTree time divergence

The data for estimated time divergence between two species were retrieved from the tool TimeTree. TimeTree provides two types of time divergence estimates; *median time* and *adjusted time*. We collected and used the *adjusted time* estimate in the unit million years ago (MYA) as evolutionary distance, since it has been calibrated to account for differences in methodology and molecular clock models across the source studies [11].

Given two species, TimeTree returns an estimated time since the two species diverged based on genetic data and fossil records. In this case, we defined evolutionary distance as the time since divergence. Table 3.2 shows an overview of the datasets, illustrating the diverse range of both broad and narrow species divergence times. For example, within the set of 135 eukaryotes, the estimated divergence time between the species within the dataset varied from 0.1 MYA to 1530 MYA. Reasonably, this should signify substantial differences between the genomes which the VLMC distance should be able to differentiate.

Organism	Domain or Kingdom	Variance in MYA
Prokaryote (149)	Bacteria (134) & Archaea (15)	0.001 - 4180
Eukaryote (135)	Animalia (44), Plantae (42) & Fungi (49)	0.1 - 1530
Eukaryote (43)	Catarrhinis (43)	0.5 - 29

Table 3.2: Summary of the used datasets and their respective evolutionary distance (from TimeTree, measured in million years ago).

A sample of collected divergence times from TimeTree based on this approach can be seen in Table 3.3. Notice that several pairs of species have the same divergence time, possibly making the training more difficult for the model.

Adjusted time divergence (MYA)	Hoolock leuconedys	Symphalangus syndactylus	Gorilla gorilla	Chlorocebus aethiops	Macaca thibetana	Semnopithecus entellus
Hoolock leuconedys	0					
Symphalangus syndactylus	8.5	0				
Gorilla gorilla	19.5	19.5	0			
Chlorocebus aethiops	28.8	28.8	28.8	0		
Macaca thibetana	28.8	28.8	28.8	12.3	0	
Semnopithecus entellus	28.8	28.8	28.8	17.8	17.8	0

Table 3.3: Time divergence in MYA for selected species, collected from TimeTree [11].

3.2.4 VLMC distance data

From the datasets in Section 3.2.1, we constructed VLMCs using Gustafsson *et al.*'s VLMC construction algorithm [16]. To get a diverse dataset for the model, we created the VLMCs by altering the VLMC parameters of minimum frequency $m \in \{25, 100\}$, maximum length of k-mers $k \in \{9, 12\}$, and the threshold value $t \in \{0.0, 3.9075\}$. As a result, a total of 8 different combinations for each genome were generated, as explained in Section 2.2.2.

To further explore the Catarrhini dataset, we generated additional combinations of VLMC parameters with larger values of k. These larger values were expected to capture more complex sequences within these more extensive genomes. In total, we constructed 24 samples for each genome, using $k \in \{9, 12, 15, 18, 21, 24\}$, $m \in \{25, 100\}$ and $t \in \{0, 3.9075\}$.

Followingly, we used Gustafsson *et al.*'s distance function, named d_v^*, to calculate the dissimilarity between each pair of VLMCs [16]. The resulting data was stored and matched with its respective evolutionary distance.

For training and testing the models, a total of 65 534 samples were generated for both E. coli and SARS-CoV-2 genomes. Similarly, 88 208 samples were generated for the set of 149 prokaryotes, 72 360 samples for the 135 eukaryotes, and 21 792 samples for the 43 Catarrhini species.

3.3 Features and deep regression model

In this section, we will focus on the architecture of our deep regression model and its features.

3.3.1 Features used in model

In total, we considered and included 8 features for the models:

- **VLMC distance**. The distance between two VLMCs is calculated using d_v^*.
- **VLMC size** $(genome\ 1)$. The sum of the length of all k-mers for the first VLMC.
- **VLMC size** $(genome\ 2)$. The sum of the length of all k-mers for the second VLMC.
- **GC content** $(genome\ 1)$. The ratio of G's and C's in the first genome.
- **GC content** $(genome\ 2)$. The ratio of G's and C's in the second genome.
- **Threshold** (t). The Kullback-Leibler threshold value.
- **Minimum frequency** (m). The minimum frequency allowed for a k-mer in the VLMC construction.
- **Maximum length** (k). The maximum length allowed for a k-mer in the VLMC construction.

The VLMC distance was the central feature of investigation for this project, owing to its ability to capture complex relationships between sequences and its correlation to evolutionary distance, see Section 2.3.2. We expected that this feature would have the most substantial influence on the models' predictability in comparison to other features.

Additionally, the sizes of the VLMCs, as described in Section 2.2.2, were included as features for each genome's VLMC. This was done since VLMCs of different sizes can have high dissimilarities, which may not be fully captured by the VLMC distance alone. Genome sequences that are larger in size have a higher probability of having repeated patterns, thus resulting in larger VLMC sizes. Also, Gustafsson has noticed that VLMC size has a nearly linear relationship with genome size, as seen in Figure 3.4 [10]. By including VLMC size as a feature, the model gains further insights into the original sequence.

As discussed in Section 2.1.4, GC-content is commonly used to analyze genomic sequences. Hence, the two genomes' respective GC-content ratios were included as

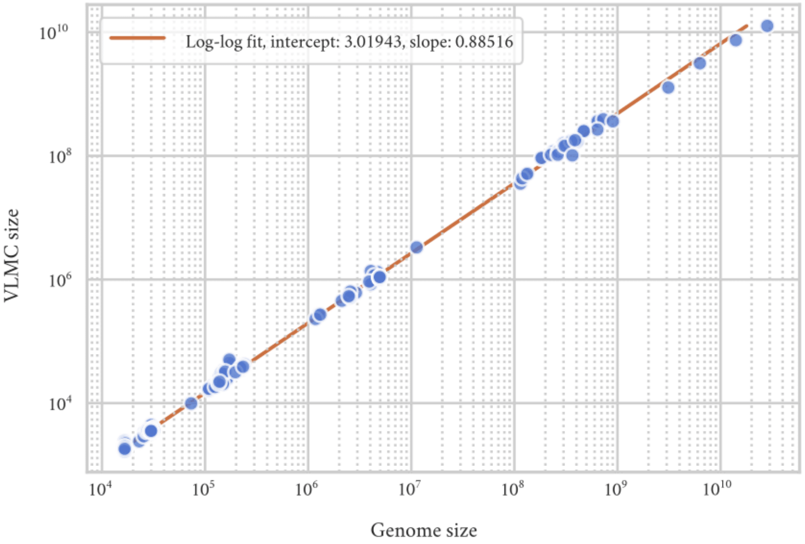


Figure 3.4: The relationship between genome size and VLMC size in a log-log fit. (Source: [10]).

features to further help the model distinguish between the two species.

To get a more general model for VLMCs, we included all three VLMC construction parameters as features to make the model independent of how the VLMCs are constructed. Those would only be utilized when we trained a general model, and not when we trained models for specific VLMC combinations.

Our data was arranged as follows:

$$
\begin{array}{cccccccc}
\textbf{VLMC}_{\textbf{dist}} & \textbf{VLMC}_{\textbf{size}} & \textbf{GC} & \boldsymbol{t} & \boldsymbol{m} & \boldsymbol{k} & & \textbf{Evolutionary distance} \\
d_v^*(x_{\text{org}}, x_1) & size_{org,1} & GC_{org,1} & 0.0 & 25 & 9 & & \text{Evo}_{\text{dist}}(\text{x}_{\text{org}}, x_1) \\
d_v^*(x_{\text{org}}, x_2) & size_{org,2} & GC_{org,2} & 0.0 & 25 & 12 & & \text{Evo}_{\text{dist}}(\text{x}_{\text{org}}, x_2) \\
d_v^*(x_{\text{org}}, x_3) & size_{org,3} & GC_{org,3} & 0.0 & 100 & 9 & & \text{Evo}_{\text{dist}}(\text{x}_{\text{org}}, x_3) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \sim & \vdots \\
d_v^*(x_{\text{org}}, x_8) & size_{org,8} & GC_{org,8} & 3.9075 & 100 & 12 & & \text{Evo}_{\text{dist}}(\text{x}_{\text{org}}, x_8) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\
d_v^*(x_{\text{org}}, x_N) & size_{org,N} & GC_{org,N} & 3.9075 & 100 & 12 & & \text{Evo}_{\text{dist}}(\text{x}_{\text{org}}, x_N)
\end{array}
$$

3.3.2 Regressor model and training

A feedforward neural network regression model (*sklearn.neural_network.MLPRegressor*) was trained and initialized in line with Tang *et al.*'s prior work [15, 32]. The features were normalized to mitigate the influence of outliers in the learning process. For the Timetree data, the target variable was normalized as well due to its exponential nature. When normalizing the data, it was important to make use of a pipeline architecture to prevent data leakage due to the MLPRegressor package having a built-in split for the validation set. The best parameter setup was evaluated using

GridSearchCV to find the optimal combination, as illustrated in the code below [32].

```python
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import GridSearchCV

model = MLPRegressor()
number_of_kfolds = 10

parameter_space = {
    'hidden_layer_sizes': [(2500,2000,1500,1000),
                           (2000,1500,1000,1000),
                           (1000,1000,1000)],
    'activation': ['relu'],
    'solver': ['adam'],
    'alpha': [0.0001, 0.001],
    'learning_rate': ['constant','adaptive'],
    'tol':[1e-6]
}

grid_search = GridSearchCV(
        mlp_regressor,
        parameter_space,
        cv=number_of_kfolds
    )

grid_search.fit(X_train, y_train)
```

Listing 3.1: Example of how the code was structured to find optimal parameters for the MLP Regressor models. Note: This example omits the normalization of the data as well as the pipeline architecture.

Training and test data were split in an 80/20 ratio to provide an independent evaluation of the model's performance on unseen data. To minimize the risk of overfitting, 10-fold cross-validation was used on the training set. This process helped to ensure that the model was robust and able to generalize well to new and unseen data.

3.4 Measuring performance

To measure the performance of our models, we compared them with a linear regression model (*sklearn.linear_model.LinearRegression*) [32]. Linear regression was chosen as a benchmark due to its simple implementation and intuitiveness. Given the known correlation between VLMC dissimilarities and the evolutionary distance, we anticipated that the linear regression model would perform reasonably well, but not fully capture the non-linear relationships between the independent variables and the evolutionary distance.

The performance of our models was based on the metrics R^2, root mean squared error (RMSE), and Spearman's ρ. R^2 was used because it is a common goodness-of-fit measure for regression models as described earlier. RMSE was selected for its sensitivity to large errors, effectively penalizing bigger discrepancies between predicted and actual values, hence handling outliers in a way that better reflects their impact on model performance. Lastly, Spearman's ρ was used because it is a good measure for the non-linear relationships between the predicted and true output values.

4

Results

4.1 Results from synthetic mutations

We begin by presenting the result from the generation of the datasets for both SARS-CoV-2 and E. coli. Subsequently, we present the results from the trained models on each genome and its performance.

4.1.1 Generation of synthetic datasets

The datasets were generated based on the virus SARS-CoV-2 and the bacterium E. coli, and their respective genomes, as shown in Figure 4.1. The graphs show the calculated dissimilarities between the original genome and the synthetically mutated genomes. The x-axis represents the VLMC dissimilarity ${d_v}^*$, and the y-axis represents the evolutionary distance.

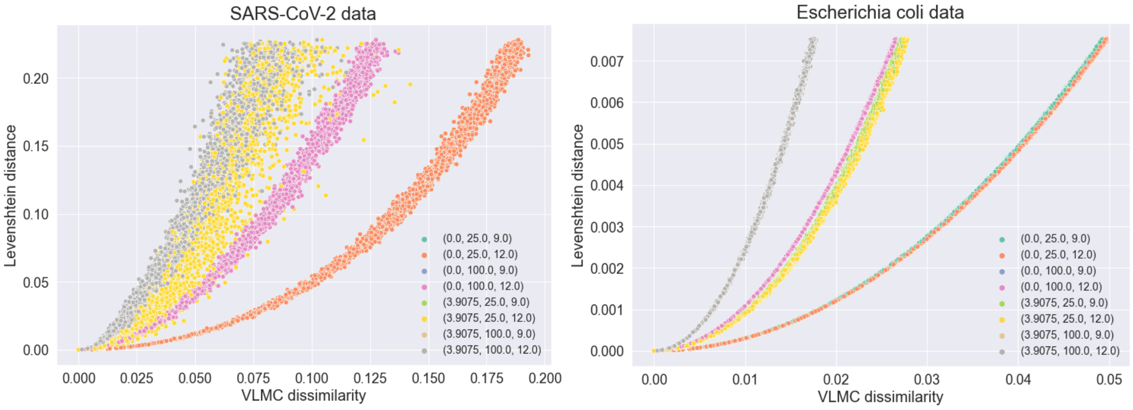


Figure 4.1: The two datasets with hue based on the three construction parameters of VLMCs in order; threshold, the minimum frequency of a k-mer allowed, and the maximum length of a k-mer.

For E. coli, the dataset indicates a consistent correlation between the VLMC dissimilarity and their respective evolutionary distance regardless of the selection of specific VLMC parameters. However, for SARS-CoV-2, it is noted that this correlation becomes increasingly less stable, characterized by increasing noise, particularly when using higher threshold values.

4.1.2 Estimating the evolutionary distance

The neural network was trained to predict the evolutionary distance based on the 6 features as described in 3.3.1. The results of the E. coli model are shown in Figure 4.2.

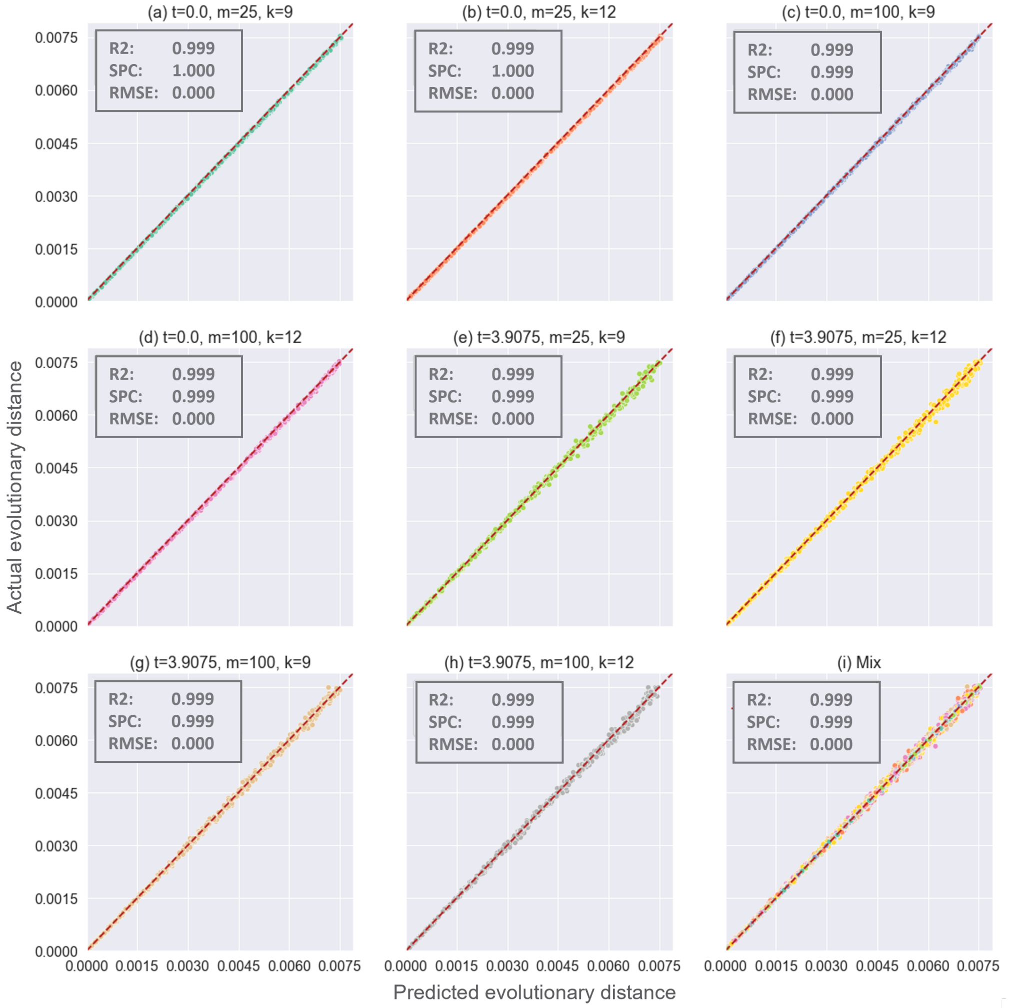


Figure 4.2: Subplots comparing the actual vs. predicted values for the MLPRegressor for different VLMC parameter combinations. (i) shows the overall performance for all combinations. The dashed red line represents the ideal relationship between actual and predicted values.

As expected from looking at Figure 4.1, the model could predict the evolutionary distance accurately for the E. coli data. Compared to the linear regressor benchmark, the neural network regressor was superior on all measures, as seen in Table 4.1.

Model	R^2	SPC	RMSE
MLP Regressor	0.99	0.99	0.00001
Linear Regressor	0.86	0.97	0.0008

Table 4.1: Performance of regressors on E. coli dataset.

For SARS-CoV-2, the results for the model are shown in Figure 4.3. The results for the different VLMC parameter combinations vary significantly more compared to the E. coli model, as was anticipated from the noisy data in Figure 4.1. However, once again, the general neural network regressor outperformed the linear regressor benchmark on all measures as seen in table 4.2.

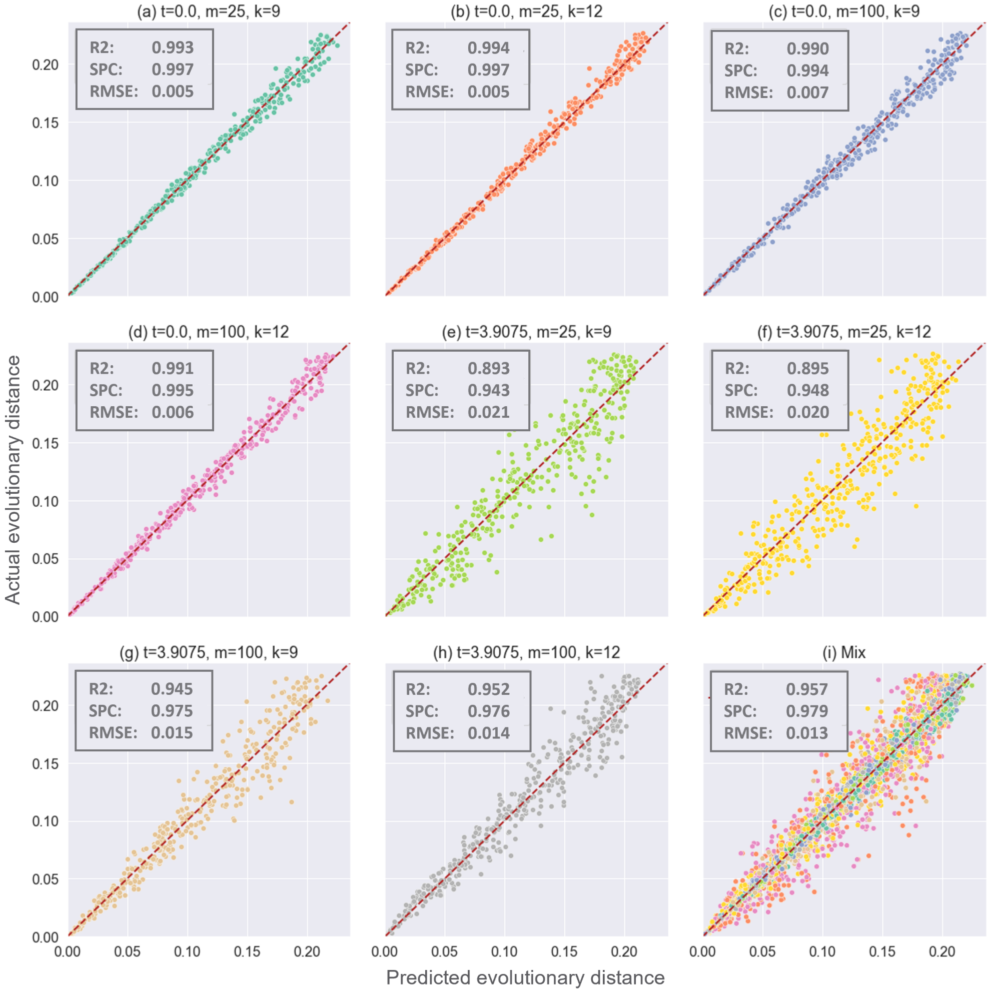


Figure 4.3: Plots comparing the actual vs. predicted values for the MLPRegressor for different VLMC parameter combinations. (i) shows the overall performance for all combinations. The dashed red line represents the ideal relationship between actual and predicted values.

Additionally, we looked at the feature weights of the neural networks for the two general MLP models, as shown in Figure 4.4 and Figure 4.5. Note that the VLMC size feature for the modified genome has a considerably larger weight for the SARS-

Model	R^2	SPC	RMSE
MLP Regressor	0.96	0.98	0.013
Linear regressor	0.84	0.95	0.025

Table 4.2: Performance of regressors on SARS-CoV-2 dataset.

CoV-2 model than for the E. coli model. Also, note that the genome size does not change for the synthetically mutated genomes, thus the linear relationship between genome size and VLMC size mentioned in 3.3.1 does not apply here. However, this result suggests that the VLMC size feature is of greater importance when the genome changes by a larger amount.

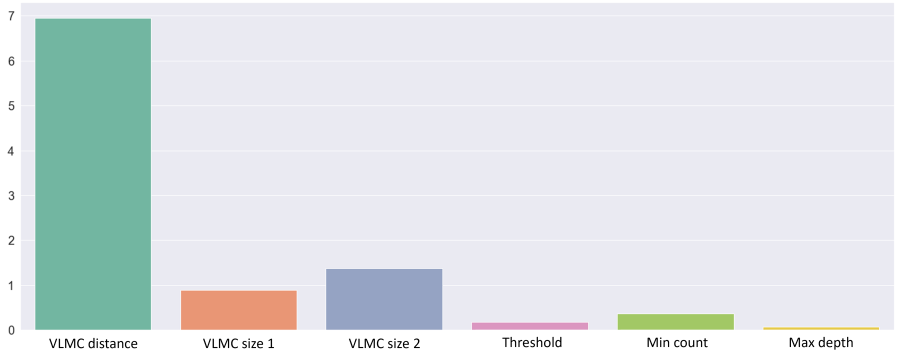


Figure 4.4: The values of the feature weights for the E. coli model, which shows the importance of the features to the model.

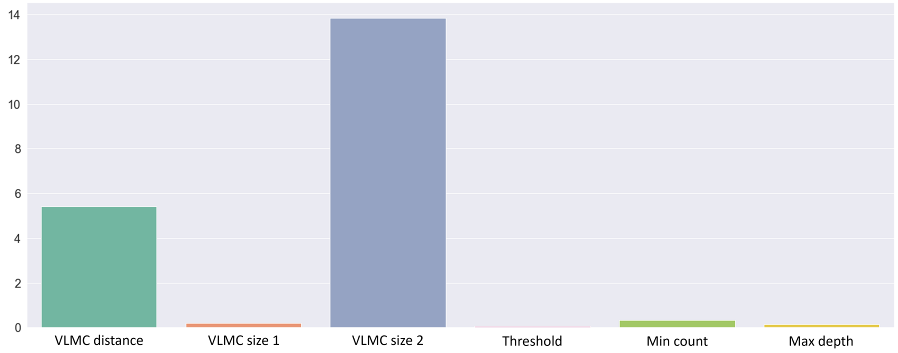


Figure 4.5: The values of the feature weights for the SARS-CoV-2 model, which shows the importance of the features to the model.

4.2 Results from the TimeTree evolutionary distance

For the TimeTree data, we begin by presenting the results from the two broad datasets of eukaryotes and prokaryotes. Subsequently, we present the results from the more narrow eukaryote dataset.

4.2.1 Results from two broad datasets

Two broad datasets of 149 prokaryotes and 135 eukaryotes were generated. A total of 8 combinations of different thresholds, minimum frequencies, and maximum lengths were tested for each dataset. Due to the scatter plot getting too cluttered using several combinations, we only show two combinations of each dataset in Figure 4.6. Due to the large size of the eukaryote dataset, which contained genomes of up to 3 billion base pairs or 3 GB in size, computational constraints limited the generation to a minimum k-mer length of 25.

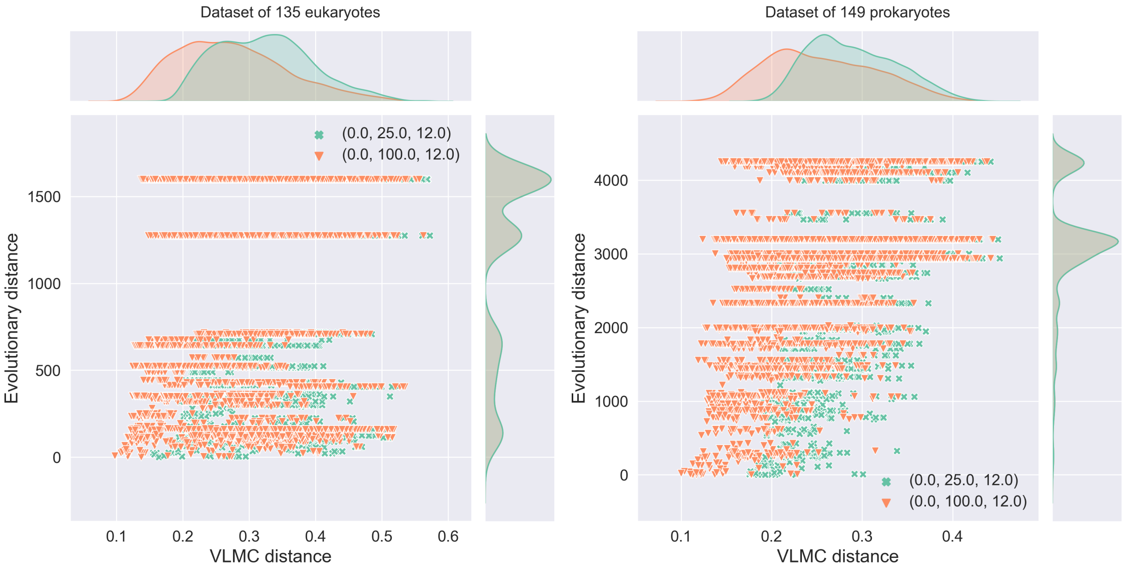


Figure 4.6: The two sets of eukaryotes and prokaryotes for fixed $t = 0$, $k = 12$, and varying $m \in \{25, 100\}$. Note how the spread of the samples' VLMC dissimilarity is greater at lower divergence times for the eukaryote set.

As seen in Figure 4.6, the eukaryote dataset showed a high degree of variability. Conversely, the prokaryote data displayed a higher degree of correlation, suggesting that a neural network model combined with additional features could yield strong predictive capability.

We also observed that the data appeared noisier with this definition of evolutionary distance as compared to the synthetically created evolutionary distance in 4.1. This can be seen in the broad horizontal lines at various levels along the y-axis in Figure 4.6. For example, the vertical density plot in the figure shows just how many data points are present around 3000 MYA. The underlying reason for this can be attributed to the concurrent divergence events occurring among several species. The concurrent divergence events among different species introduce complexities for the training data, as the VLMCs show a variation on a per-genome basis.

Two general MLP Regressors were trained on the two respective datasets. In Table 4.3, the performance of three different models on the prokaryote dataset is shown.

As anticipated from the motivation in 3.3.1 and the result in 4.1.1, excluding the VLMC size feature made the MLP Regressor perform significantly worse. Additionally, in Figure 4.7, one can see that the model performed quite evenly among the different VLMC parameter combinations.

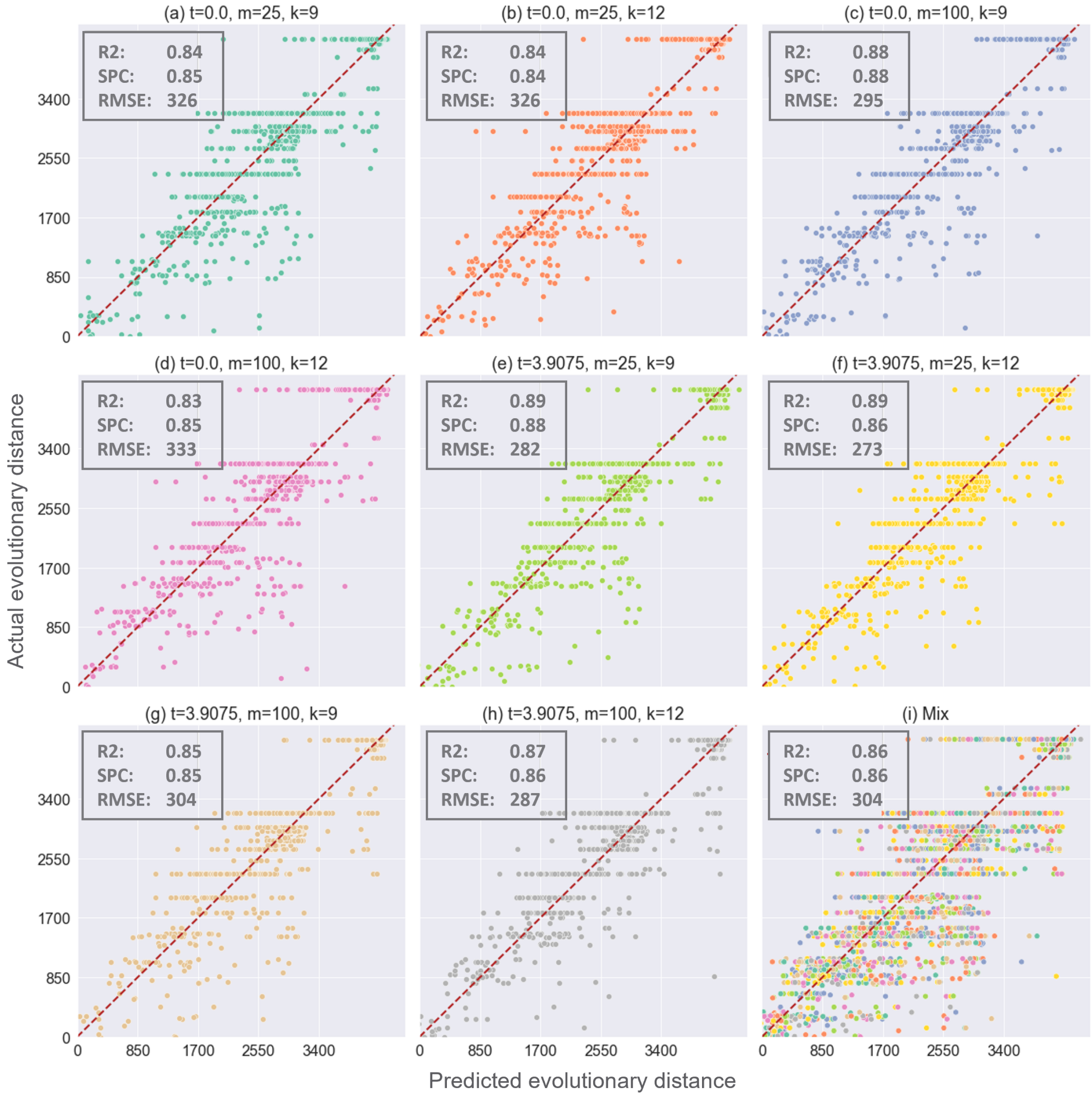


Figure 4.7: The prokaryote model's performance for the eight different combinations of VLMC parameters and the overall performance.

Model	$\mathbf{R^2}$	**SPC**	**RMSE**
MLP Regressor	0.86	0.86	304
MLP Regressor (excl. VLMC size features)	0.61	0.69	514
Linear regressor	0.18	0.36	741

Table 4.3: Performance of the regressors on the 149 prokaryote dataset.

Further, it is worth mentioning that we generated VLMC sets for the prokaryote genomes with $k > 12$ and noticed that these were nearly identical to the data where $k = 12$, meaning larger k-mers rarely occur in these genomes. Thus, no models were produced to test these sets.

In table 4.4, the results for the model on the eukaryote dataset are shown. Once again, the model excluding the VLMC size feature performed worse. It should also be noted that the performance is worse for this dataset in general. Additionally, it is evident that this model exhibits a higher performance in predicting samples with a value of $k = 12$ when compared to samples with $k = 9$, as depicted in Figure 4.8.

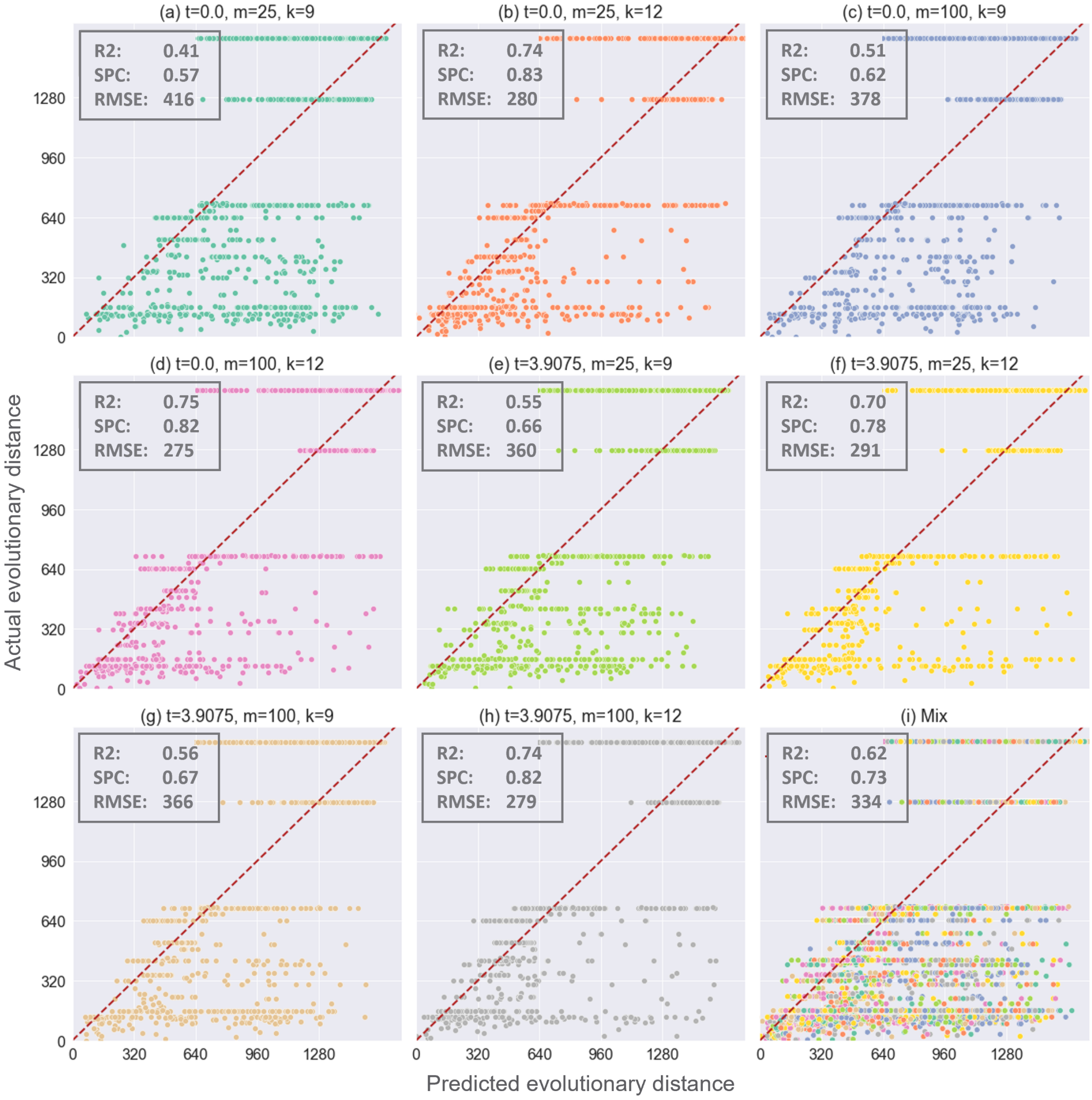


Figure 4.8: The eukaryotes model's performance for the eight different combinations of VLMC parameters and the overall performance.

Given the noisy input data, the general model that was trained on multiple VLMC parameters had poor performance in absolute terms. The model yielded an R^2 score of 0.62, a Spearman's ρ of 0.73, and a Root Mean Square Error (RMSE) of

Model	$\mathbf{R^2}$	SPC	RMSE
MLP Regressor	0.62	0.73	334
MLP Regressor (excl. VLMC size features)	0.37	0.54	431
Linear regressor	0.09	0.27	516

Table 4.4: Performance of the regressors on the 135 eukaryote dataset.

334 MYA. Using the specific VLMC parameters $k = 12$, $m = 100$, and $t = 0$, the model demonstrated an R^2 score of $0 - 75$, a Spearman's ρ of 0.82, and an RMSE of 275 MYA.

Subsequently, due to the poor performance, we wanted to examine whether we could reach better performance with a model trained on a dataset consisting of more closely related species in terms of evolutionary distance.

4.2.2 Results from the dataset of 43 Catarrhini's

A dataset was generated consisting of 43 species of Catarrhinis, a parvorder within the Primate order. In comparison to the larger dataset of 135 eukaryotes, which had a divergence time of up to 1530 MYA, this Catarrhini dataset covers a narrower range of up to 29 MYA. Consequently, the narrower range should mean significantly fewer genome mutations.

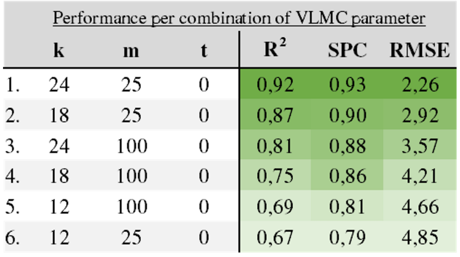
For this dataset, we generated and examined several values of k. The motivation for exploring various several values stemmed from observing the eukaryote model's superior performance in relation to larger k values. Therefore, as explained in Section 3.2.4, we considered values of $k \in \{9, 12, 15, 18, 21, 24\}$.

Two model configurations were developed, each having different feature sets. The first general model included all features, while the second excluded the VLMC size feature. The performance of these two models, as well as a benchmark linear regressor, is presented in Table 4.5. Note that the performance of the general model clearly outperforms that of the broad eukaryote model above.

Model	$\mathbf{R^2}$	SPC	RMSE
MLP Regressor	0.86	0.90	3.15
MLP Regressor (excl. VLMC size features)	0.71	0.84	4.45
Linear regressor	0.44	0.69	6.30

Table 4.5: Performance of the regressors on the 43 Catarrhini dataset.

A more extensive analysis of 6 combinations, as illustrated in Figure 4.9, demonstrates that the combinations with $k = 24$ and $k = 18$, along with $m = 25$, yield superior performance compared to other combinations. A comparison between $k = 9$ and $k = 24$ is provided in Figure 4.10, which presents a density plot comparing the two datasets with fixed m and t. Note how $k = 24$ is more densely packed, suggesting that a higher value of k corresponds with a simpler pattern to identify and explains the superior performance compared to lower k.



Performance per combination of VLMC parameter

	k	m	t	R^2	SPC	RMSE
1.	24	25	0	0,92	0,93	2,26
2.	18	25	0	0,87	0,90	2,92
3.	24	100	0	0,81	0,88	3,57
4.	18	100	0	0,75	0,86	4,21
5.	12	100	0	0,69	0,81	4,66
6.	12	25	0	0,67	0,79	4,85

Figure 4.9: Performance of different VLMC parameter combinations for the dataset of 43 Catarrhinis, ranked by the best performance.

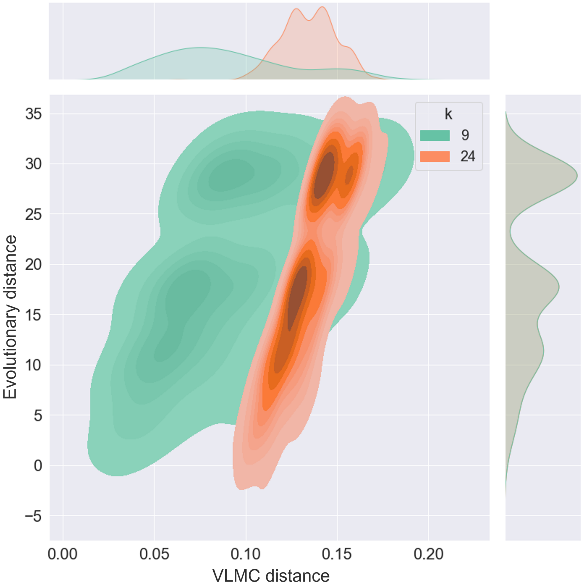


Figure 4.10: A density plot of $k = 9$ and $k = 24$ for the Catarrhini dataset with fixed $m = 25$ and $t = 0$.

In Figure 4.11, we show the edge cases of $k \in \{9, 24\}$ of the general model to illustrate the difference in performance between these two combinations of k. The model performs superior for $k = 24$ compared to $k = 9$.

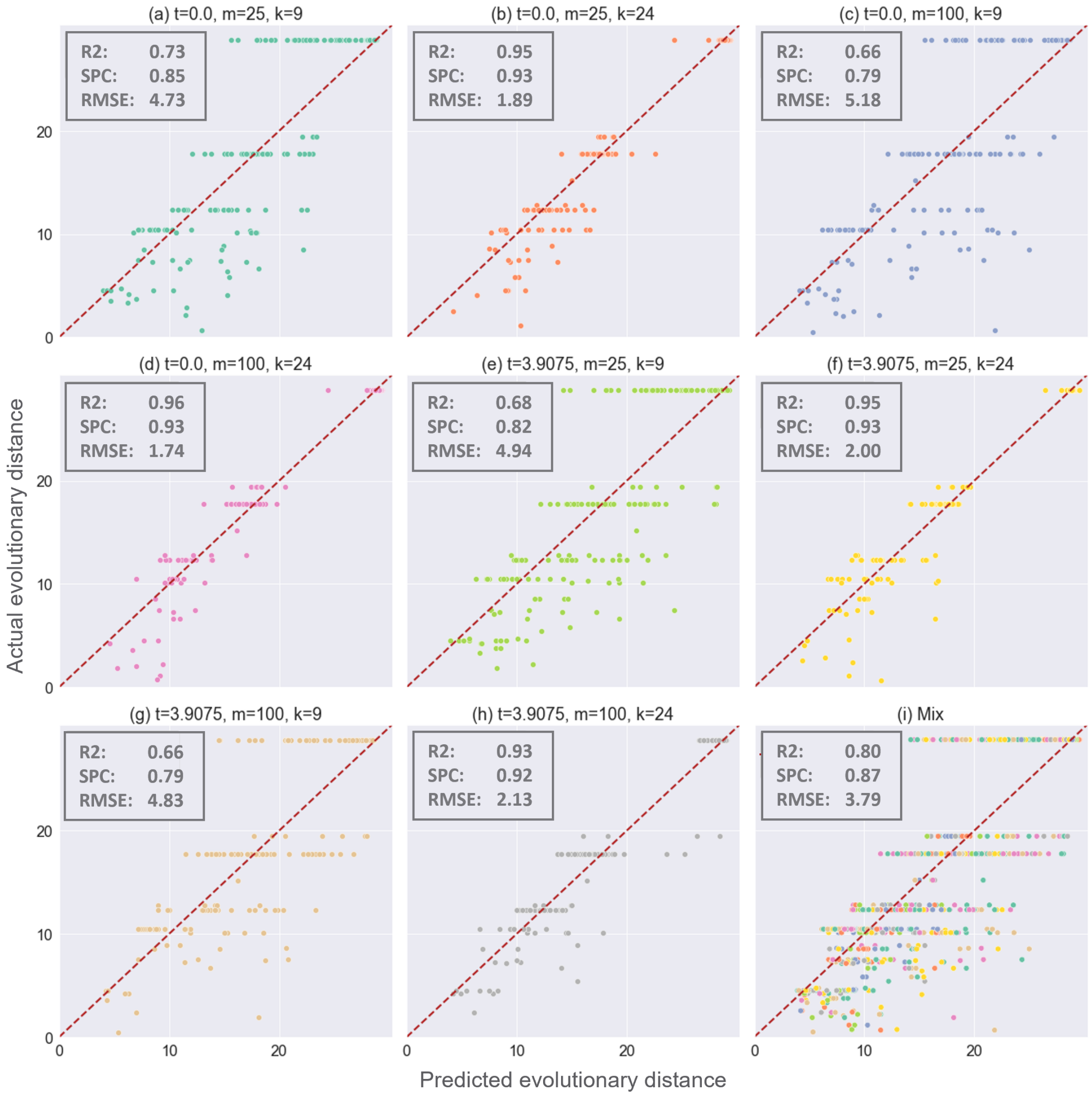


Figure 4.11: The Catarrhini model's performance for the eight different edge cases where $k \in \{9, 24\}$ and the overall performance of this.

Finally, when inspecting the feature weights of the model including VLMC size, shown in Figure 4.12, it becomes evident why VLMC size makes a great difference in performance.

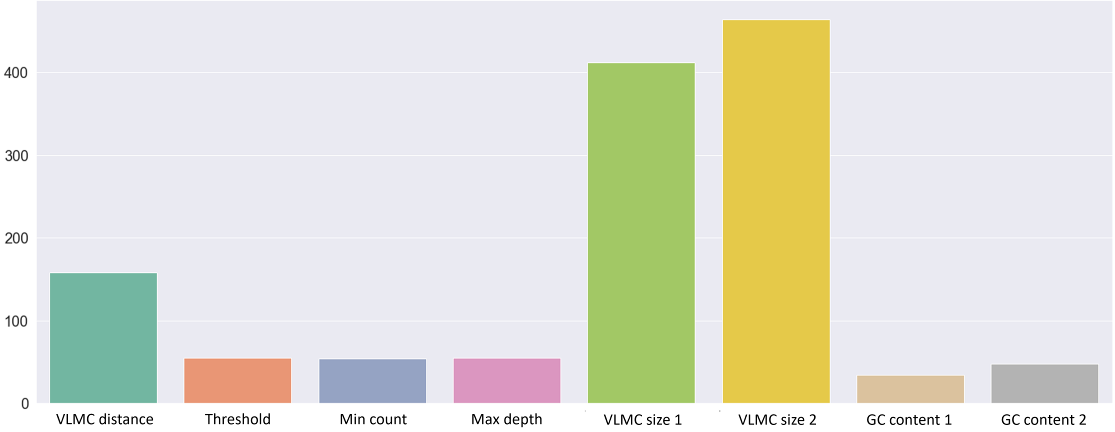


Figure 4.12: The feature weights for the Catarrhini model when VLMC sizes are included. Note that both of the VLMC size features are heavily weighted.

4.2.3 Results from a general model vs. a specific model

Given the best VLMC combination for the Catarrhini model ($t = 0$, $m = 100$, and $k = 24$) we compared the general model (trained on multiple combinations) to a model solely trained on the specific combination. Interestingly, we found that the general model outperformed the specifically trained model, as can be seen in Figure 4.13 and 4.14.

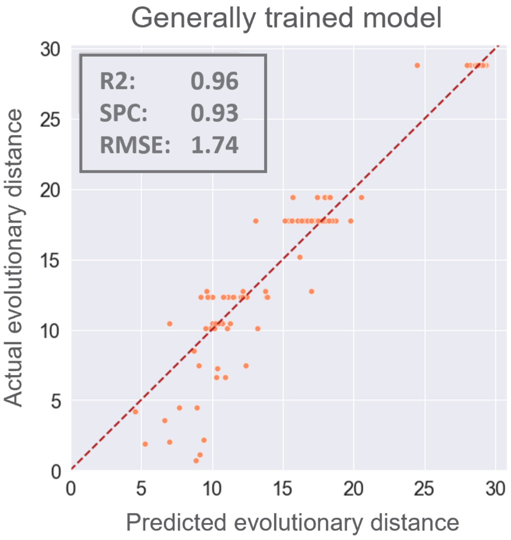


Figure 4.13: Generally trained model's predictions on the VLMC combination $t = 0$, $m = 100$, and $k = 24$.

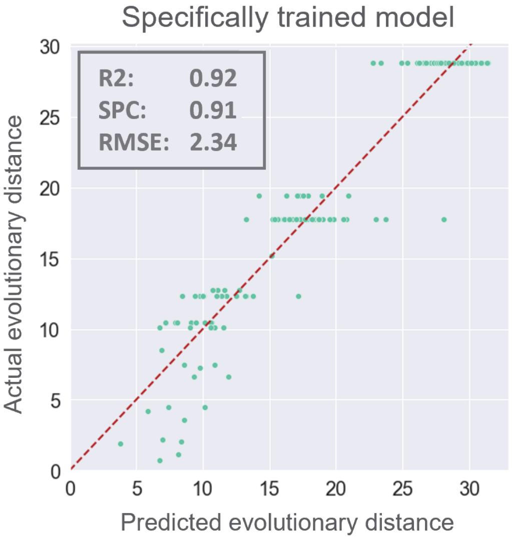


Figure 4.14: Specifically trained model's predictions on the VLMC combination $t = 0$, $m = 100$, and $k = 24$.

5

Discussion and Conclusion

5.1 Discussion

In this section, we look into the findings of our study, focusing primarily on our neural network models' performance in predicting evolutionary distance using synthetic mutation data and TimeTree data. We discuss the various influential factors affecting the models' performance, the implications of these findings and how they may provide insights into future work.

5.1.1 Mutation data

The results, as presented in Section 4.1, show that our neural network models for the synthetically created mutations of both E. coli and SARS-CoV-2 outperformed the linear regressor benchmark on evaluation metrics; R^2, Spearman, and RMSE. While both models performed impressively, the E. coli model generally had superior performance, especially with regard to RMSE.

One critical factor that might have affected the performance of each model is the difference in genome size between the two genomes. The E. coli genome is significantly larger than that of SARS-CoV-2 (1.4 MB vs. 0.01 MB), which influences the mutation limit of E.coli due to computational restrictions. For E.coli, mutations were constrained to a maximum of 1% of the entire genome, while mutations for SARS-CoV-2 were constrained to 22.5% of its genome. This discrepancy led to a greater degree of variation in the VLMC construction for SARS-CoV-2 and as a result, the VLMC size feature was of greater importance for the SARS-CoV-2 model, as demonstrated in Figure 4.4 and 4.5. Although, the increased VLMC complexity and size also apparently affects the performance negatively, compared to the E. coli model.

RMSE, representing the standard deviation of prediction errors, is another vital factor to discuss. Both E.coli and SARS-CoV-2 models achieved low RMSE values, indicating that the model's predictions were close to the actual evolutionary distances. Specifically, the E. coli achieved an RMSE of 0.0001 and the SARS-CoV-2 model achieved a value of 0.013. These values were significantly lower than the RMSE values for the linear regressor.

The general results of the models demonstrate the efficacy in capturing the relationship between evolutionary distance (in the form of Levenshtein distance) and VLMC distance combined with VLMC size. This is particularly effective when using specific VLMC parameter values, mainly setting the pruning threshold t to 0. However, a lower threshold comes with the trade-off of increased computational demands.

5.1.2 TimeTree data

Several interesting points arose from the results of the TimeTree data models. From the two broad sets of prokaryotes and eukaryotes, we found that there was a significant difference between the accuracy of the two models. The performance of the prokaryote model had an $R^2 = 0.86$, compared to the eukaryote model with $R^2 = 0.62$. However, both models had a substantial margin of error in estimating evolutionary distance, as indicated by a high RMSE of 300 RMSE. Interestingly, we noticed a decrease in performance by $20 - 40\%$ when the VLMC size feature was dropped during model training.

By focusing our analysis from the broad eukaryote dataset that ranged up to 1530 MYA, to a more narrow dataset of 43 Catarrhinis with a range of up to 29 MYA, the model showed significantly better performance. The R^2 value increased from 0.62 to 0.86 and the RMSE decreased from 304 to 3.15, indicating that the model's predictions were considerably more precise within this narrower range.

First, let's look into the reasons behind the superior performance of the prokaryote model compared to the eukaryote model. If you observe Figure 4.6, you will notice broader vertical lines for the eukaryote dataset in contrast to the prokaryote dataset. The fundamental reason here is that prokaryotes, being single-cell organisms, have genomes of limited size which we believe leads to fewer unique k-mers. Consequently, genome pairs with the same evolutionary distance have less variability in their VLMC dissimilarity. Moreover, the smaller size of prokaryotic genomes makes the presence of longer k-mers less frequent at higher minimum frequencies, as compared to eukaryotic genomes. This understanding reinforces the better performance of the prokaryote model.

Second, we observed that the VLMC construction parameters are pivotal for the models' performance. We found that lower m values led to better performance in the prokaryotic model, while k values > 12 were less relevant as these longer k-mers did not appear often enough. However, excessively small m values (e.g. $m = 2$) resulted in poor performance. We hypothesize that this could be because including k-mers that seldom appear makes the VLMCs too generalized, decreasing the uniqueness of the VLMCs and making the VLMC size less informative.

Furthermore, higher k values substantially improved the accuracy of the eukaryote model. We believe that longer k-mers are more unique and hence more representative of species groups. Thus, it helps the model to determine if two genomes are close in time or not.

Lastly, including the VLMC size as an input feature to the neural network greatly contributes to model performance, especially for eukaryotes. This is likely related to the larger genome sizes of eukaryotes, which makes the VLMC size an effective tool for the model to deduce the difference between VLMCs. Moreover, the significance of the VLMC size feature likely increases with k, as the VLMC becomes more distinct and its size becomes more unique with larger k values (more like a fingerprint).

5.1.3 Generally trained model vs. specifically trained model

As was shown in the results, we noticed that a generally trained model (based on 8 different VLMC combinations) performed better on a specific VLMC combination set, compared to a model trained specifically for that combination set. We believe this could be attributed to the use of data augmentation. This suggests that training the model on additional samples through more VLMC combinations may be beneficial. An interesting difference between these two models, besides the performance, was the feature weights. The generally trained model had VLMC size heavily weighted, as discussed earlier, whereas the specifically trained model weighted VLMC distance the most.

5.1.4 Future work

It is important to note that the results obtained from the synthetic mutation data are derived from a controlled synthetic data set. To further validate these results and identify areas for model improvement, it would be beneficial to provide additional tests using different alignment-based methods for calculating evolutionary distance.

Furthermore, upon reflecting on the outcomes of our broad prokaryote model and the Catarrhini model, we believe that VLMCs have the potential to be far more accurate when examining specific subgroups of prokaryotes. Therefore, we suggest exploring VLMCs' potential in niche bacterial groups, similar to our investigation with Catarrhini but for prokaryotes.

Another interesting approach would be to develop a classification model that predicts the family, order, parvorder, and other taxonomic classifications to which a genome belongs. This has previously been done through clustering methods, but not through deep neural networks [27].

5.2 Conclusion

In conclusion, this thesis has explored VLMCs and d_v^*'s potential in predicting two different definitions of evolutionary distance through deep regression. For the Levenshtein distance, the models could accurately predict the number of synthetic mutations made on two distinct genomes, proving a deep regression approach based on VLMCs and d_v^* can be used as an alignment-free method. For the TimeTree divergence, the results were promising and the accuracy of predicting a more narrow dataset was better than for predicting a broader dataset. However, it is suggested that one could further explore more specific datasets and additional features to increase the performance of these models.

Bibliography

[1] N. H. G. R. I. (NHGRI), “Acgt.” https://www.genome.gov/genetics-glossary/acgt, 2023. Accessed: 2023-05-16.

[2] I. Merlu, “Pleurotus cornucopiae,” 2011. License: CC BY-SA 3.0.

[3] Luisovalle, “Cacao (theobroma cacao),” 2007. License: CC BY-SA 4.0.

[4] J. S. Charles, “Rock grayling (hipparchia semele) male,” 2016. License: CC BY-SA 4.0.

[5] S. Duckworth, “Bukhara deer stag at speyside wildlife park,” 2008. Accessed: 2023-05-24.

[6] H. Hoyer, “At the psychiatrist,” 2008. License: CC BY-SA 2.0.

[7] J. Friedl, “Male guinea baboon in nuremberg zoo,” 2005. License: CC BY-SA 2.0.

[8] D. David, “Proboscis monkey in borneo,” 2010. License: CC BY-SA 2.0.

[9] K. Bakie, “Orangutan cincinnati zoo,” 2005. License: CC BY-SA 2.5.

[10] J. Gustafsson, “Private communication,” 2023.

[11] S. Kumar, G. Stecher, M. Suleski, and S. B. Hedges, “Timetree 5: An expanded resource for species divergence times,” *Molecular Biology and Evolution*, 2022.

[12] P. Bühlmann and A. Wyner, “Variable length markov chains,” *The Annals of Statistics*, vol. 27, 01 2004.

[13] D. Dalevi, D. Dubhashi, and M. Hermansson, “A new order estimator for fixed and variable length markov models with applications to dna sequence similarity,” *Statistical Applications in Genetics and Molecular Biology*, vol. 5, no. 1, 2006.

[14] D. Dalevi, D. Dubhashi, and M. Hermansson, “Bayesian classifiers for detecting hgt using fixed and variable order markov models of genomic signatures,” *Bioinformatics*, vol. 22, pp. 517–522, 01 2006.

[15] K. Tang, J. Ren, and F. Sun, “Afann: Bias adjustment for alignment-free sequence comparison based on sequencing data using neural network regression,” *Genome Biology*, vol. 20, 12 2019.

[16] J. Gustafsson, S. Edwards, A. Schliep, and P. Norberg, “Estimating phylogenies from raw sequencing reads using variable-length markov chains,”

[17] J. Geoghegan and E. Holmes, “The phylogenomics of evolving virus virulence,” *Nature Reviews Genetics*, vol. 19, 10 2018.

[18] M. Holmudden, “Virus attenuation by genome-wide alterations of genomic signatures,” 2015.

[19] A. Muto and S. Osawa, “The guanine and cytosine content of genomic dna and bacterial evolution,” *Proceedings of the National Academy of Sciences*, vol. 84, no. 1, pp. 166–169, 1987.

[20] N. C. for Biotechnology Information (NCBI), "National center for biotechnology information." https://www.ncbi.nlm.nih.gov/, 1988. Internet. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; [cited 2017 Apr 06].

[21] P. A. Gagniuc, *Markov Chains: From Theory to Implementation and Experimentation*. John Wiley & Sons, 2017.

[22] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[23] M. H. Schulz, D. Weese, T. Rausch, A. Döring, K. Reinert, and M. Vingron, "Fast and adaptive variable order markov chain construction," in *Algorithms in Bioinformatics* (K. A. Crandall and J. Lagergren, eds.), (Berlin, Heidelberg), pp. 306–317, Springer Berlin Heidelberg, 2008.

[24] S. Lab, "dvstar," 2023. Accessed: 2023-03-29.

[25] W. M. Berger B and Y. Y. Levenshtein, "Distance, sequence comparison and biological database search," *IEEE Trans Inf Theory*, no. 67, p. 3287–3294, 2021.

[26] L. Bromham and D. Penny, "The modern molecular clock," *Nature reviews. Genetics*, vol. 4, pp. 216–24, 04 2003.

[27] J. Gustafsson and E. Norlander, "Clustering genomic signatures a new distance measure for variable length markov chains," Master's thesis, Chalmers University of Technology, 2018.

[28] D. C. Torney, C. Burks, D. Davison, and K. M. Sirotkin, "Computation of d2: a measure of sequence dissimilarity," *Computers and DNA: The Proceedings of the Interface between Computation Science and Nucleic Acid Sequencing Workshop*, 1990.

[29] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. Springer, 2013.

[30] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 ed., 2009.

[31] Z. Lab, "What is fasta format?." https://zhanglab.ccmb.med.umich.edu/FASTA/, n.d.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

A

Genomes from the dataset of 43 Catarrhinis

Table A.1: A list of all genomes utilized in the 43 Catarrhini's dataset. It includes the assembly accession numbers and corresponding genome sizes for each genome.

Species	**Assembly number**	**Size (MB)**
Cercocebus atys	GCF_000955945.1	2 751,6
Cercopithecus mona	GCA_014849445.1	2 803,1
Cercopithecus neglectus	GCA_004027615.1	3 388,7
Chlorocebus aethiops	GCA_024741045.1	2 726,4
Chlorocebus sabaeus	GCF_015252025.1	2 837,7
Colobus angolensis	GCF_000951035.1	2 869,7
Erythrocebus patas	GCA_024740955.1	2 990,8
Gorilla gorilla	GCF_029281585.1	3 476,8
Hoolock leuconedys	GCA_024740915.1	2 689,5
Hylobates moloch	GCA_009828535.3	2 752,5
Hylobates pileatus	GCA_021498465.1	2 753,4
Lophocebus aterrimus	GCA_023783235.1	2 803,9
Macaca arctoides	GCA_021188215.1	2 743,5
Macaca assamensis	GCA_024740895.1	2 664,6
Macaca cyclopis	GCA_026956025.1	2 748,1
Macaca fuscata	GCA_003118495.1	2 837,5
Macaca nemestrina	GCF_000956065.1	2 848,5
Macaca nigra	GCA_928851695.1	2 821,7
Macaca silenus	GCA_024740855.1	2 683,4
Macaca thibetana	GCF_024542745.1	2 725,4
Mandrillus leucophaeus	GCA_023783495.1	2 958,3
Mandrillus sphinx	GCA_023783085.1	2 739,0
Miopithecus talapoin	GCA_028551445.1	2 935,6
Nasalis larvatus	GCA_004027105.1	2 939,5
Nomascus leucogenys	GCF_006542625.1	2 746,4
Nomascus siki	GCA_024740865.1	2 687,6
Pan paniscus	GCF_029289425.1	3 159,1
Papio anubis	GCA_008728515.2	2 772,4
Papio hamadryas	GCA_024740875.1	2 949,5

Continued on next page

Table A.1 – continued from previous page

Specie	Assembly number	Size (MB)
Papio papio	GCA_028645565.1	2 849,4
Piliocolobus tephrosceles	GCF_002776525.5	2 894,5
Pongo abelii	GCA_028885685.1	3 250,1
Pongo pygmaeus	GCF_028885625.1	3 137,8
Pygathrix nemaeus	GCA_004024825.1	3 247,1
Pygathrix nigripes	GCA_024740835.1	2 796,9
Rhinopithecus bieti	GCF_001698545.2	2 885,2
Rhinopithecus roxellana	GCA_024741075.1	2 934,4
Rhinopithecus strykeri	GCA_023764705.1	2 833,7
Semnopithecus entellus	GCA_004025065.1	3 038,2
Symphalangus syndactylus	GCA_028878085.1	3 073,5
Theropithecus gelada	GCA_028533075.1	2 792,2
Trachypithecus francoisi	GCA_009764325.1	2 776,8
Trachypithecus phayrei crepuscula	GCA_023762245.1	2 772,4

B

Genomes from the dataset of 135 eukaryotes

Table B.1: A list of all genomes utilized in the 149 prokaryotes dataset. It includes the assembly accession numbers and corresponding genome sizes for each genome.

Species	**Assembly number**	**Size (MB)**
Abramis brama	GCA_022829085.1	1 030,7
Acer negundo	GCA_025594385.1	427,2
Aegilops longissima	GCA_904067125.1	5 598,7
Aegilops sharonensis	GCA_904067115.1	6 484,7
Aegilops tauschii	GCF_002575655.2	4 089,0
Agriopis marginaria	GCA_932305915.1	483,7
Antechinus flavipes	GCF_016432865.1	3 082,9
Ascochyta rabiei	GCA_004011695.2	39,5
Aspergillus luchuensis	GCA_016865315.1	36,0
Blumeria graminis	GCA_024363405.1	136,5
Boehmeria	GCA_021020685.1	260,9
Bombyx mori	GCA_027497135.1	444,6
Brachypodium distachyon	GCF_000005505.3	262,0
Buddleja alternifolia	GCA_019426215.1	824,4
Cairina moschata	GCA_018104995.1	1 080,5
Callinectes sapidus	GCA_027123505.1	964,1
Calonectria leucothoes	GCA_002179835.1	61,2
Camarhynchus parvulus	GCA_902806625.1	1 015,6
Camelus dromedarius	GCF_000803125.2	2 097,6
Candida albicans	GCA_029931695.1	13,8
Candida dubliniensis	GCA_000647575.1	14,1
Cannabis sativa	GCA_029168945.1	846,0
Capsicum chinense	GCA_026120095.1	2 983,8
Ceratopteris richardii	GCA_020310875.1	7 206,6
Cervus hanglu	GCA_010411085.1	2 506,7
Clitopilus hobsonii	GCA_015708445.2	35,7
Coffea canephora	GCA_900059795.1	550,3
Colletotrichum higginsianum	GCA_023705605.1	49,0
Corymbia calophylla	GCA_014182845.1	381,3

Continued on next page

Table B.1 – continued from previous page

Specie	Assembly number	Size (MB)
Cucurbita argyrosperma	GCA_004115005.2	246,4
Danio aesculapii	GCA_903798145.1	1 334,0
Drosophila pseudoobscura	GCA_023515705.1	157,7
Drosophila teissieri	GCF_016746235.2	144,4
Eremothecium cymbalariae	GCF_000235365.1	9,3
Erigeron canadensis	GCF_010389155.1	411,8
Erithacus rubecula	GCA_903797595.2	1 049,5
Eucalyptus melliodora	GCA_004368105.3	617,6
Exobasidium	GCA_026210745.1	17,0
Flammulina velutipes	GCA_945909995.2	36,8
Fragaria iinumae	GCA_009720345.1	232,3
Fraxinus pennsylvanica	GCA_912172775.2	730,8
Fulvia fulva	GCF_020509005.1	64,9
Fundulus heteroclitus	GCF_011125445.2	1 162,2
Fusarium circinatum	GCA_024514935.1	45,2
Fusarium poae	GCF_019609905.1	42,5
Fusarium pseudograminearum	GCA_024752415.1	35,7
Gambusia affinis	GCF_019740435.1	656,8
Gossypium anomalum	GCA_025698475.1	1 161,4
Gossypium gossypioides	GCA_013467495.1	674,6
Gossypium hirsutum	GCA_024704785.1	2 226,9
Gouania willdenowi	GCA_900634775.2	905,0
Gynochthodes officinalis	GCA_020080225.1	468,2
Hericium erinaceus	GCA_016906435.1	39,8
Hermetia illucens	GCF_905115235.1	970,4
Hipparchia semele	GCA_933228805.2	389,5
Hydra vulgaris	GCA_024232925.1	791,2
Hypsizygus marmoreus	GCA_023718595.1	42,2
Kazachstania africana	GCF_000304475.1	10,8
Kluyveromyces lactis	GCA_947297865.1	10,4
Kluyveromyces marxianus	GCA_029873725.1	10,6
Lachancea dasiensis	GCA_900074725.1	10,3
Lachancea mirantina	GCA_900074745.1	9,8
Lachancea nothofagi	GCA_900074755.1	10,9
Lachancea thermotolerans	GCA_027604975.1	10,0
Lagenaria siceraria	GCA_002890555.2	287,6
Leersia perrieri	GCA_000325765.2	257,5
Leptidea sinapis	GCA_949711765.1	662,0
Limenitis camilla	GCA_905147385.1	420,2
Luffa acutangula	GCA_012295215.1	711,0
Lycium barbarum	GCA_019175385.1	1 612,3
Macaria	GCA_927399415.1	380,2
Marasmius oreades	GCA_025815895.1	42,9

Continued on next page

Table B.1 – continued from previous page

Specie	Assembly number	Size (MB)
Mayetiola destructor	GCA_001014435.1	182,0
Melampsora medusae	GCA_002157035.1	81,9
Meloidogyne hapla	GCA_000172435.1	51,5
Metopolophium dirhodum	GCA_019925205.1	464,3
Metschnikowia reukaufii	GCA_003401635.1	15,0
Morchella rufobrunnea	GCA_024713595.1	55,3
Motacilla alba	GCA_015832205.1	1 035,9
Nematolebias whitei	GCF_014905685.2	1 178,8
Neofusicoccum parvum	GCA_029169195.1	42,5
Nephelium	GCA_021234005.1	395,2
Nezara viridula	GCA_928085145.1	1 144,4
Nyssa sinensis	GCA_008638375.1	967,0
Oncorhynchus mykiss	GCA_029834435.1	2 261,2
Ovis aries	GCA_024256505.1	2 537,8
Paracanthobrama guichenoti	GCA_018749465.1	1 051,0
Paulownia fortunei	GCA_019321725.1	494,3
Phocoena sinus	GCF_008692025.1	2 289,9
Phoenix dactylifera	GCA_000181215.3	747,0
Pholidichthys leucotaenia	GCA_020510965.1	5 824,8
Phycomyces blakesleeanus	GCF_001638985.1	52,1
Pichia kudriavzevii	GCA_029581905.1	10,4
Pipistrellus pipistrellus	GCA_903992545.1	1 702,8
Pleurotus cornucopiae	GCA_019677325.2	31,3
Plodia interpunctella	GCF_027563975.1	281,3
Pneumocystis carinii	GCF_001477545.1	7,4
Pochonia chlamydosporia	GCA_000411695.4	42,7
Podospora	GCA_017654855.1	33,5
Prunus davidiana	GCA_020226225.1	235,5
Pyricularia oryzae	GCA_026261775.1	39,6
Pyricularia pennisetigena	GCF_004337985.1	47,4
Pyrrhoderma noxium	GCA_016618065.1	30,5
Rhododendron ovatum	GCA_019656835.1	530,8
Saccharomyces paradoxus	GCA_918281175.1	11,7
Saccharomycopsis malanga	GCA_004014935.1	16,4
Salix suchowensis	GCA_029030765.1	369,0
Sceloporus tristichus	GCA_016801065.1	1 754,2
Sesamum indicum	GCA_027475695.1	267,7
Solanum pennellii	GCF_001406875.1	894,6
Taeniopygia guttata	GCF_003957565.2	1 020,0
Takifugu rubripes	GCA_901000725.3	370,9
Talaromyces pinophilus	GCA_027569545.1	35,3
Teleopsis dalmanni	GCA_002237135.4	603,0
Tetrapisispora phaffii	GCF_000236905.1	11,7

Continued on next page

Table B.1 – continued from previous page

Specie	Assembly number	Size (MB)
Theobroma cacao	GCF_000208745.1	313,8
Thermothelomyces thermophilus	GCF_000226095.1	37,4
Theropithecus gelada	GCA_028533075.1	2 792,2
Thoracobombus	GCA_905332965.1	297,0
Thyatira batis	GCA_905147785.1	304,0
Torulaspora delbrueckii	GCA_937863135.1	8,9
Trametes hirsuta	GCA_001302255.2	36,2
Trichothecium roseum	GCA_026184415.1	32,6
Triticum dicoccoides	GCF_002162155.2	10 331,4
Typha latifolia	GCA_020740505.1	207,0
Urochloa ruziziensis	GCA_015476505.1	583,9
Ustilago maydis	GCA_928722245.1	19,0
Vaccinium macrocarpon	GCA_022606695.1	468,2
Vitis rotundifolia	GCA_022557335.1	380,3
Xenentodon cancila	GCA_014839995.1	710,9
Xenopus laevis	GCF_017654675.1	2 648,1
Yarrowia lipolytica	GCA_029531965.1	19,8
Zingiber officinale	GCA_011317585.2	2 984,1
Ziziphus jujuba	GCF_020796205.1	391,8
Zymoseptoria tritici	GCA_029873855.1	38,3

C

Genomes from the dataset of 149 prokaryotes

Table C.1: A list of all genomes utilized in the 135 eukaryotes dataset. It includes the assembly accession numbers and corresponding genome sizes for each genome.

Species	Assembly number	Size (MB)
Acaryochloris marina	GCF_024347835.1	8.1
Acidothermus cellulolyticus	GCF_023512095.1	2.4
Acinetobacter baumannii	GCA_030056375.1	3.8
Actinoplanes missouriensis	GCF_000284295.1	8.5
Aerococcus viridans	GCF_029023765.1	2.1
Alkalihalobacillus krulwichiae	GCF_002109385.1	4.4
Ammylolactobacillus amylophilus	GCF_006540265.1	1.6
Anabaena cylindrica	GCF_014696465.1	6.8
Anaerostipes hadrus	GCA_948466995.1	3.0
Anoxybacillus amylolyticus	GCF_004346755.1	3.1
Arcanobacterium haemolyticum	GCA_021532125.1	1.9
Archaeoglobus sulfaticallidus	GCF_000385565.1	2.0
Azospirillum thiophilum	GCF_001305595.1	7.4
Bacillus thuringiensis	GCA_029712825.1	6.0
Bacteroides caccae	GCA_949948875.1	5.3
Bartonella quintana	GCF_024731665.1	1.5
Bifidobacterium actinocoloniiforme	GCF_001263395.1	1.8
Borreliella valaisiana	GCF_018282055.1	1.2
Brachyspira hyodysenteriae	GCF_027708025.1	3.0
Brachyspira murdochii	GCA_020722965.1	3.1
Brevibacillus brevis	GCA_029958365.1	6.5
Brevibacillus laterosporus	GCF_029478705.1	5.2
Brevibacterium aurantiacum	GCF_014897605.1	3.9
Caldivirga maquilingensis	GCF_000018305.1	2.0
Carboxydothermus hydrogenoformans	GCF_000012865.1	2.3
Castellaniella defragrans	GCF_017848875.1	3.8
Cellulosilyticum lentocellum	GCF_000178835.2	4.6
Chitinophaga pinensis	GCA_943328055.1	8.8
Chlamydia felis	GCF_000009945.1	1.1

Continued on next page

Table C.1 – continued from previous page

Specie	Assembly number	Size (MB)
Chlamydia muridarum	GCA_937877055.1	1.0
Chloroflexus aggregans	GCA_002877815.1	4.5
Clostridium argentinense	GCA_018371285.1	4.6
Clostridium butyricum	GCF_030035785.1	4.5
Clostridium formicaceticum	GCF_002080475.1	4.4
Clostridium saccharobutylicum	GCF_014231195.1	4.9
Corynebacterium singulare	GCF_022346285.1	2.7
Corynebacterium ureicelerivorans	GCF_946221375.1	2.3
Cupriavidus necator	GCA_946479875.1	7.2
Cyanobacterium aponinum	GCF_014697365.1	4.0
Dactylococcopsis salina	GCF_000317615.1	3.7
Dehalococcoides mccartyi	GCA_937863645.1	1.3
Desulfomicrobium orale	GCF_001553625.1	2.7
Desulforapulum autotrophicum	GCA_020723285.1	5.5
Desulfurococcus mucosus	GCF_001006085.1	1.3
Dokdonella koreensis	GCF_001632775.1	4.3
Dyella thiooxydans	GCF_001641285.1	4.1
Eggerthella lenta	GCF_028616345.1	3.4
Erwinia amylovora	GCF_028623185.1	3.7
Eubacterium limosum	GCF_028861895.1	4.3
Ferriphaselus amnicola	GCF_000974685.2	2.6
Gottschalkia purinilytica	GCF_001190785.1	3.3
Halioglobus pacificus	GCF_014652275.1	3.3
Halobacterium salinarum	GCF_021504045.1	2.4
Halomonas campaniensis	GCF_014193375.1	3.8
Haloterrigena turkmenica	GCF_001483125.1	5.3
Hydrogenobacter thermophilus	GCA_025062355.1	1.7
Hyphomicrobium nitrativorans	GCF_000503895.1	3.5
Ignicoccus islandicus	GCF_001481685.1	1.4
Intestinimonas butyriciproducens	GCA_948444905.1	3.1
Jonesia denitrificans	GCF_019048465.1	2.7
Kangiella geojedonensis	GCF_000981765.1	2.4
Kocuria palustris	GCF_029876575.1	2.8
Kushneria marisflavi	GCF_003610515.1	3.5
Lacrimispora saccharolytica	GCF_024460435.1	4.5
Leptolyngbya boryana	GCF_022372495.1	6.6
Lutibacter litoralis	GCF_014646675.1	3.5
Macrococcus caseolyticus	GCF_024205865.1	2.1
Marinovum algicola	GCA_950075755.1	5.2
Mesotoga prima	GCF_018432665.1	2.9
Methanobrevibacter ruminantium	GCA_029012245.1	2.8
Methanocaldococcus fervens	GCF_000023985.1	1.5
Methanohalophilus halophilus	GCF_003722055.1	2.0

Continued on next page

Table C.1 – continued from previous page

Specie	Assembly number	Size (MB)
Methanohalophilus mahii	GCF_000025865.1	1.9
Methanosphaera stadtmanae	GCA_905200445.1	1.7
Microbulbifer agarilyticus	GCF_020171345.1	4.1
Micromonospora aurantiaca	GCF_020995455.1	6.8
Mycobacterium haemophilum	GCF_025823035.1	4.1
Mycolicibacterium gilvum	GCF_025821885.1	5.6
Myxococcus xanthus	GCF_025739275.1	8.8
Nakamurella multipartita	GCF_000024365.1	5.9
Natrinema pellirubrum	GCF_000337635.1	4.2
Natronomonas pharaonis	GCF_000026045.1	2.7
Neisseria lactamica	GCA_945878035.1	2.1
Neorhizobium galegae	GCF_024384685.1	6.2
Neorickettsia sennetsu	GCF_000013165.1	0.8
Nitrobacter hamburgensis	GCF_000013885.1	4.8
Nocardia farcinica	GCF_029869805.1	6.1
Nocardiopsis dassonvillei	GCF_024181525.1	6.3
Nonlabens tegetincola	GCF_002954355.1	2.7
Nostoc punctiforme	GCF_014698495.1	8.8
Novosphingobium resinovorum	GCF_027922145.1	6.7
Octadecabacter arcticus	GCF_000155735.2	5.3
Oleidesulfovibrio alaskensis	GCF_016756815.1	3.6
Oscillatoria nigro-viridis	GCF_000317475.1	8.0
Paenarthrobacter aurescens	GCF_025421775.1	4.2
Pandoraea pnomenusa	GCF_017848955.1	5.2
Paracholeplasma brassicae	GCF_000967915.1	1.8
Parascardovia denticolens	GCF_938044075.1	1.8
Pediococcus pentosaceus	GCF_029823395.1	1.8
Photobacterium gaetbulicola	GCF_003025515.1	5.7
Pimelobacter simplex	GCF_024662375.1	5.4
Pluralibacter gergoviae	GCA_029768725.1	5.2
Priestia megaterium	GCF_029958225.1	5.4
Pseudoalteromonas translucida	GCF_001465295.1	3.7
Pseudomonas citronellolis	GCF_029335335.1	6.7
Pseudomonas fulva	GCF_029841105.1	5.0
Ralstonia mannitolilytica	GCF_027319325.1	4.9
Rhodococcus fascians	GCA_030063685.1	5.3
Rhodoferax antarcticus	GCF_025961535.1	3.9
Rhodoferax ferrireducens	GCF_003347515.1	4.8
Rhodospirillum centenum	GCF_000016185.1	4.2
Roseiflexus castenholzii	GCF_002877835.1	5.5
Ruegeria pomeroyi	GCF_021405555.1	4.4
Salinispora tropica	GCF_000620345.1	5.0
Serratia liquefaciens	GCF_028621345.1	5.2

Continued on next page

Table C.1 – continued from previous page

Specie	Assembly number	Size (MB)
Shewanella amazonensis	GCF_000015245.1	4.2
Shewanella denitrificans	GCF_000013765.1	4.4
Sphingomonas koreensis	GCF_028550725.1	4.7
Sphingomonas melonis	GCA_947374955.1	3.8
Spirochaeta thermophila	GCF_000184345.1	2.5
Spiroplasma chrysopicola	GCF_000400935.1	1.1
Spiroplasma eriocheiris	GCF_002028345.1	1.3
Staphylococcus aureus	GCA_030053715.1	2.7
Staphylococcus pseudintermedius	GCA_029983135.1	2.5
Stenotrophomonas acidaminiphila	GCF_030020345.1	3.7
Streptococcus iniae	GCA_030020405.1	2.0
Streptomyces albus	GCF_030011775.1	7.8
Syntrophothermus lipocalidus	GCF_012840045.1	2.3
Taylorella equigenitalis	GCF_029948185.1	1.7
Thalassolituus oleivorans	GCF_029210095.1	3.8
Thauera humireducens	GCF_001051995.2	4.0
Thermacetogenium phaeum	GCA_001508215.1	2.8
Thermobispora bispora	GCF_019686255.1	4.2
Thermococcus nautili	GCF_000585495.1	1.9
Thermococcus sibiricus	GCA_001508065.1	1.8
Thermocrinis albus	GCF_000025605.1	1.5
Thermus aquaticus	GCF_001399775.1	2.3
Thermus brockianus	GCF_022846375.1	2.3
Thermus scotoductus	GCF_027319405.1	2.2
Thioalkalivibrio sulfidiphilus	GCF_000377945.1	3.4
Thiomicrospira cyclica	GCA_022736205.1	1.9
Thiomonas intermedia	GCA_022730375.1	3.2
Tolumonas auensis	GCF_000023065.1	3.4
Treponema brennaborense	GCA_023666275.1	3.0
Treponema putidum	GCF_024401155.1	2.7
Vibrio alginolyticus	GCA_029228015.1	5.0
Vibrio vulnificus	GCF_029990625.1	4.9
Weissella ceti	GCF_025908515.1	1.5
Xanthomonas sacchari	GCF_029761895.1	4.8