Politecnico di Torino

Politecnico di Torino

Data Science and Engineering

A.a. 2024/2025

Graduation Session December 2025

Automatic detection of fitness shifts in pathogen phylogenies using contrastive learning

Supervisors:

Roberta Bardini
Gabriele Marino
Alexander Zarebski
Stefano Di Carlo
Alessandro Savino

Candidate:

Mario Capobianco

Abstract

The continual emergence and extinction of pathogen lineages, driven by factors including immune escape, environmental changes, or differences in transmissibility, poses major challenges for public health. Detecting lineages with an increased fitness is critical for understanding epidemiological shifts and guiding targeted interventions. Existing approaches for detecting fitness changes among lineages, such as PhyloWave, extract summary statistics from trees and use them to identify lineages with different evolutionary dynamics. However, PhyloWave depends on substantial domain knowledge and thresholds that require manual fine-tuning—often relying on expert judgment or arbitrary choices—which limits its scalability and robustness across different pathogens.

To overcome the limitations of existing approaches, we integrate contrastive representation learning with phylogenetic modeling to implement a generalization of the multi-type birth-death (MTBD) model in which mutation events alter lineage-specific transmission rates. These simulations generate training and testing data that capture a wide range of fitness scenarios. Building on this foundation, we design a supervised learning strategy for community detection in phylogenies, where recursive neural networks learn clade representations. By optimizing a contrastive loss, our model is encouraged to learn separate representations of lineages undergoing distinct fitness dynamics. Claderepresentations are input to a classification head, where similarity between each tree node and its parent acts as a continuous metric of relatedness. Higher similarity indicates a greater likelihood of shared evolutionary properties, enabling identification of variants with fitness changes along the tree. This approach eliminates dependence on ad hoc parameterization and establishes a principled and scalable methodological framework for monitoring the evolutionary fitness of circulating lineages, with potential applications across viral and bacterial pathogens.

Acknowledgements

This work would not have been possible without the commitment and dedication of my amazing supervisors: Gabriele, Roberta and Alex. Gabriele helped me from conceptualization to development, and thanks to his ideas and support, even the most difficult steps seemed manageable. Roberta provided invaluable supervision and very generous funding. Alex, with his creative and colorful ideas, has always been available and kind from conceptualization to development.
This thesis is as much theirs as it is mine.

A special thank you goes to my friends and family.
To Peppe, who welcomed me into his life with kindness and naiveté.
To Maurizio, our friendship has seen so many ups and downs these past two years, but only you and I can understand certain burdens of life and yet we have very different ways of dealing with them.
To Nicole, who, with her sharp irony, has always been a good friend to me.
To Fiona, with whom I can talk about anything, from the most trivial things to the most difficult problems.
To Michi, with his childhood dreams.
To Natan, I know you are destined to do great things.
To Roberto, the one true friend from elementary school all the way to the university desks.
To Alessio, with his intelligence and dedication.
To my parents and sister, who supported me both economically and emotionally for the largest part of my life, and, unfortunately, will not be here to celebrate with me, I wish that someday you may be free from all the undeserved difficulties you have encountered.
To my loving nana, who, looking at the photograph of my grandfather and me, prays for me every single day.

To my wonderful nono, who welcomed me with open arms and praised me for my accomplishments, how much I'm missing the nights we played cards and I wish you could come back.
To my relatives, who invited me everywhere they could.

Finally, to the most important person of my life, Martina, who taught me what it means to love and to care for someone other than yourself. From barely speaking to each other to living together in less than a year is surely a great step, but I never questioned this decision. You are the only person who can deal with me when I'm just overwhelmed by every kind of emotion. You turned the darkest time of my life into the most beautiful. Thank you.

Table of Contents

List of Figures VII

1 Introduction 1
- 1.1 Context: Genomic epidemiology . . . 1
- 1.2 Technical domain: Phylogenetic and phylodynamic modeling . . . 1
- 1.3 The challenge: Detecting evolutionary communities . . . 2
- 1.4 Existing solutions: Deep learning as a surrogate model . . . 3
- 1.5 Proposed work: A deep-learning framework for clade detection . . . 3
- 1.6 Thesis structure . . . 4

2 Background 5
- 2.1 Birth-Death models . . . 5
  - 2.1.1 Phylogenetic Trees . . . 8
  - 2.1.2 Multi-type birth-death models . . . 13
  - 2.1.3 Community Detection problem . . . 16
- 2.2 Neural Networks for Tree Data . . . 18
  - 2.2.1 How to use Neural Networks for Surrogate Deep Learning . 19
  - 2.2.2 Summary statistic representations . . . 20
  - 2.2.3 Compact Bijective Ladderized Vector (CBLV) representation 20
  - 2.2.4 Graph neural network approaches . . . 21
  - 2.2.5 Towards recursive architectures . . . 22
- 2.3 Contrastive Learning . . . 22
  - 2.3.1 Foundations of Contrastive Learning . . . 23
  - 2.3.2 Contrastive Learning in Hierarchical Data . . . 23
  - 2.3.3 Community Detection as Classification . . . 24
- 2.4 Analysis of the state-of-the-art . . . 25
  - 2.4.1 Variant Hunters . . . 25
  - 2.4.2 PhyloWave and other approaches . . . 26
  - 2.4.3 Limitations of current paradigms . . . 26

3 Materials & Methods 31
3.1 Contribution I - Adapted recursive neural networks 31
3.2 Contribution II - Contrastive Loss for Community Detection 38
3.2.1 Smoothed Contrastive Loss 38
3.3 Materials 39
3.3.1 MTBD-based mutation model 39
3.3.2 Simulation data 43
3.3.3 Dataset 45
3.3.4 Empirical data 47
3.4 Analysis & NN training 48
3.4.1 Validation of training data & Empirical data 49
3.4.2 Network Architecture 51
3.4.3 Optimization 52
3.5 Implementation & Computation 53
3.5.1 Model requirements 53

4 Results 55
4.1 Dataset Metrics Analysis 55
4.2 Evaluation Metrics 57
4.2.1 Contrastive Loss 58
4.2.2 Clustering Metrics 58
4.2.3 Clustering Results on Simulated Data 59
4.2.4 Interpretation 59
4.3 Network validation 59
4.4 Inference on Empirical data 64

5 Discussion & Conclusions 70
5.1 Discussion 70
5.2 Conclusions and Future Work 71

Bibliography 73

List of Figures

2.1 Expected number of species under a birth-death with the number of lineages $n = 1000$: **top line** $\lambda - \psi > 0$, **middle line** $\lambda - \psi = 0$, **bottom line** $\lambda - \psi < 0$. Figure taken from [16] 7

2.2 Examples of three different phylogenetic trees. **A**: the true phylogenetic tree, the tree obtained observing each lineage evolving over time. **B**: the extant-species tree, the same tree considering only the lineages still alive today. **C**: a partially sampled phylogenetic tree. A subset of lineages is sampled. Figure taken from [16] 8

2.3 Trees sharing the same tree topology. The clades and branching order remain unchanged, even though the drawings differ in orientation and branch lengths. Figure taken from [16] 9

2.4 Examples of phylogenetic trees that share the same overall tree shape (A and B) and one with a different shape (C). Trees A and B display an identical branching structure once tip labels are ignored: both exhibit an unbalanced configuration in which most lineages belong to one major clade. Tree C instead shows a more balanced division of lineages, illustrating how tree shape captures the distribution of descendants across major splits independently of the specific taxa involved. Figure taken from [16] 10

2.5 Example of four different tree structures. **a**: caterpillar tree, the most unbalanced tree, with $I_S = 35$, $I_S, norm = 1$ $I_C = 21$, $I_C, norm = 1$. **b**: fully symmetric bifurcating tree, with $I_S = 24$, $I_S, norm = 0.59$ $I_C = I_C, norm = 0$ **c**: star tree, with $I_S = 8$, $I_S, norm = 0$ **d**: clone tree of the lung tumor CRUK0065 in the TRACERx cohort. Nodes represented by empty circles correspond to extinct clones, and the diameters of other nodes are proportional to the corresponding clone population sizes. Figure taken from [17] 12

2.6 Example of LTT plot: red lines correspond to transmission events and are linked from the phylogenetic tree to the LTT plot. Figure taken from [16] 13

2.7 **a** birth-death model (BD), **b** birth-death model with Exposed-Infectious individuals (BDEI), **c** birth-death model with Super-Spreading (BDSS) [10] 15

2.8 The three colored groups (green, red, and blue) represent three monophyletic clades. Each clade corresponds to a subtree defined by a single ancestral node and all of its descendants, highlighting distinct evolutionary lineages within the phylogeny. Image adapted from the Digital Atlas of Ancient Life (CC-BY 4.0). 17

2.9 PhyloDeep pipeline. Tree representations: a (i), simulated binary trees. The simulations were encoded into two representations, either a (ii–v) with CBLV or a (vi) with summary statistics (SS). CBLV is obtained through a (ii) ladderization and a (iii) an inorder tree traversal. a (iv), an input matrix in which the information on internal nodes and leaves is separated into two rows. a (v), this matrix is padded with zeros so that the matrices for all simulations have the size of largest simulation matrices. SS consists of a (vi), a set of 98 statistics: 83 published in Saulnier et al., 14 on transmission chains and 1 on tree size. The information on sampling probability is added to both representations. b Neural networks are trained on these representations to estimate parameter values or to select the underlying model. For SS, b (i), a deep feed-forward neural network (FFNN) of funnel shape . For the CBLV representation, b (ii), convolutional neural networks (CNN). The CNN is added on top of the FFNN. Figure taken from [10] 28

2.10 Example of a phylogenetic tree partitioned into three monophyletic clades (blue, green, red). Each colored region highlights a group of taxa sharing a recent common ancestor, illustrating how clades naturally define positive pairs (within the same color) and negative pairs (across different colors) for contrastive learning. Image adapted from the Wikipedia page "Clade" (`https://en.wikipedia.org/wiki/Clade`), licensed under CC BY-SA 4.0. 29

2.11 Phylowave pipeline. a, Schematics describing the principles of index computation. From left to right: example of a time-resolved phylogenetic tree with a background population (grey) and an emerging lineage (green); pairwise distance distribution from terminal node A, or terminal node B, respectively, to the rest of the population, with the dashed blue line denoting the geometric weighting; and expected index dynamics over time (see Methods for details). b–e, For each pathogen, SARS-CoV-2 (b), influenza A/H3N2 (H3N2; c), B.pertussis (d) and M.tuberculosis (e) the index dynamics computed at each node (terminal or internal) is presented. Colors represent the different lineages identified by their different index dynamics. Figure taken from [9] 30

3.1 Example of a phylogenetic tree in which each internal node is assigned a vector representation computed by the MLP together with its branch length, except for the root, where only the final encoded vector is shown. 32

3.2 Illustration of the encoding step for internal nodes: for each node, the vectors corresponding to its left and right children (with respective branch lengths) are averaged, and the resulting mean vector is passed through the MLP to produce the internal representations e_5 and e_6. 33

3.3 Example of a phylogenetic tree in which the internal nodes highlighted in red represent the base nodes selected according to the minimum–leaves criterion. For illustration purposes, the figure uses small hyperparameter values (e.g. minimum leaves per cluster and encoding dimension), which are not the ones used in the actual implementation. 36

3.4 Example of the CBLV representations assigned to the nodes directly below the base nodes identified in the previous step. Each red vector shows the CBLV encoding padded to dimension $E+1$, assigned only to the immediate children of the base nodes, while the rest of the tree is processed recursively by the neural network. 37

3.5 Plot of the smoothed contrastive loss showing its three components: the pull loss (green), which penalizes small distances for similar-anchor pairs; the push loss (red), which penalizes distances below the margin m for dissimilar-anchor pairs; and the total loss (blue), given by their weighted combination as defined in Eq. (3.10). The vertical dotted line marks the margin m. 40

3.6 Illustration of the full evolutionary tree (top) and the corresponding sampled tree (bottom).Red segments mark the occurrence and propagation of a mutation, while eye symbols indicate sampling events. Only the sampled lineages and the minimal structure connecting them are retained in the reconstructed tree. 41

3.7 Portion of a simulated tree. In red, nodes with no mutation are represented. Each color identifies a new mutation, where the birth rate is increased. 42

4.1 Distribution of mean pairwise patristic distances across simulated phylogenies. The histogram shows the distribution of the mean pairwise patristic distance computed over 100 simulated phylogenies generated under the BD-MUT model. The x-axis reports the average pairwise patristic distance for each tree, while the y-axis indicates how many trees fall within each bin. Vertical colored lines mark the corresponding values for the empirical datasets used in this thesis, Pertussis, H3N2, SARS-CoV-2, and TB, with their mean distances reported in parentheses. The empirical trees fall within (or close to) the simulated range of values, indicating that the simulated training data adequately match the evolutionary scale of real pathogen phylogenies. 56

4.2 Distribution of Sackin index values across simulated phylogenies. The histogram reports the distribution of the Sackin index for 100 simulated phylogenies, a classical measure of tree imbalance where higher values indicate more unbalanced branching structures. The x-axis shows the Sackin index of each simulated tree, while the y-axis represents the number of trees falling into each bin. Vertical colored lines correspond to the Sackin index of the empirical phylogenies used in this thesis, Pertussis, H3N2, SARS-CoV-2, and TB, along with their index values shown in parentheses. The empirical Sackin indices lie within or near the simulated distribution, indicating that the simulated training trees capture a comparable range of topological imbalance to that observed in real pathogen phylogenies. 57

4.3 Elbow heuristic for selecting the number of clusters in k-means clustering. The plot shows how the **inertia**, the within-cluster sum of squared distances between each point and the centroid of its assigned cluster, changes as a function of the number of clusters k. The x-axis reports the chosen value of k, while the y-axis reports the corresponding inertia. Increasing k always decreases inertia, because clusters become smaller and better fit the data. However, the rate of decrease is not uniform: after an initial steep drop, the curve gradually flattens. The *elbow point* marks the value of k where adding more clusters provides only marginal reductions in inertia. This point offers a practical trade-off between underfitting (too few clusters) and overfitting (too many), and is therefore used as a heuristic to select an appropriate number of clusters. 60

4.4 Example of validation tree: comparison between mutation values, true cluster labels, embedding-based similarity, and predicted labels for a representative simulated phylogeny. Top: true mutation coloring (left) and corresponding ground-truth cluster assignments with logged birth for each corresponding mutation in the legend (right). Bottom: similarity map derived from the learned node embeddings, with a histogram showing the similarity scores (left), and model-predicted cluster labels (right). The strong correspondence between predicted and true labels indicates that the encoder successfully captures the hierarchical structure of the tree and reconstructs meaningful evolutionary clusters. 61

4.5 Second validation tree example: this phylogeny contains no mutation-derived clusters in the ground truth. Top: true mutation coloring (left), and the corresponding true cluster labels (right), showing a single mutation value across the entire tree, where all nodes belong to the same cluster due to the absence of valid mutation events. Bottom: similarity map derived from the learned node embeddings (left), with a log-scaled histogram of cosine similarity values in the inset, and the predicted labels (right). As expected, the network assigns all nodes to a single cluster. 63

4.6 Inference results for the H3N2 phylogeny (subsampled from 2022). Top row: the true clade assignments derived from metadata (right) and the corresponding mutation-based coloring (left), displaying the numerous closely related sublineages that characterize recent H3N2 seasonal evolution. Bottom row: the similarity map computed from Phyloscope's learned embeddings (left), with a log-scaled histogram of cosine similarities in the inset, and the predicted cluster labels produced by the model (right). Phyloscope successfully recovers the major circulating clades, including the prominent groups `2a`, `2a.1`, and `2a.3`, and preserves the deeper structural backbone of the tree. Because this tree contains many small, shallow sublineages, the minimum cluster-size threshold was increased to 25 tips to avoid fragmentation into numerous tiny clusters. Despite merging some of the closest sublineages, the model provides a coherent reconstruction of the main evolutionary groups present in the dataset. 65
4.7 Inference results for the Pertussis phylogeny (2009–2011 subsample). Top row: true clade assignments derived from metadata (right) and the corresponding mutation-based coloring (left) after minimum leaves per cluster grouping. Bottom row: similarity map computed from the learned node embeddings (left), including a log-scaled histogram of cosine similarities in the inset, and the predicted cluster labels returned by Phyloscope (right). Two of the three major clades (`ptxP1/fim3-1` and `ptxP3/fim3-2`) are successfully recovered, with predicted clusters that match the true structure of the tree. 66
4.8 Inference results for the SARS-CoV-2 phylogeny (subsampled from 2023). Top row: mutation-based coloring (left) and true clade assignments (right), showing the dominant Omicron sublineages present in the dataset. Bottom row: similarity map derived from the learned node embeddings (left), including a log-scaled histogram of cosine similarities in the inset, and the predicted cluster labels obtained from Phyloscope (right). 67
4.9 Inference results for the Mycobacterium tuberculosis (TB) phylogeny. Top row: mutation-based coloring (left) and true clade assignments (right), showing the major Euro-American sublineages present in the dataset. Bottom row: similarity map produced by the learned node embeddings (left), including a log-scaled histogram of cosine similarities, and the resulting predicted cluster labels from Phyloscope (right). 69

Chapter 1

Introduction

1.1 Context: Genomic epidemiology

The large-scale sequencing of pathogen genomes has changed how we study infectious diseases. By combining genomic and epidemiological data, an approach known as **genomic epidemiology**, researchers can reconstruct how pathogens spread, identify new variants, and monitor epidemics in real time [1, 2].

Modern sequencing programs now produce very large datasets. Each genome captures a snapshot of the pathogen's evolutionary history, and by analyzing many genomes together it becomes possible to reconstruct how transmission and viral diversity change over time [3]. This genomic revolution has greatly improved public health responses by enabling the early detection of variants with increased transmissibility or immune escape [4]. However, the size and complexity of these data also create new challenges, motivating the need for scalable tools that can automatically detect meaningful evolutionary patterns.

1.2 Technical domain: Phylogenetic and phylodynamic modeling

To study evolutionary relationships among pathogen genomes, data are usually represented as **phylogenies** or **phylogenetic trees**: branching trees in which leaves represent sampled genomes and internal nodes represent transmission events [5]. A phylogenetic tree is the reconstruction of the pathogen's history upon sampling. Within a phylogeny:

- a **lineage** is a chain of ancestor–descendant nodes that follows the evolutionary history of a group of samples;

- a **clade** is a subtree formed by an ancestor and all of its descendants, representing a monophyletic group [1].

Phylogenetics focuses on reconstructing these trees from genetic sequences. **Phylodynamics** goes further by interpreting phylogenies using population and epidemiological models, allowing researchers to infer changes in transmission, growth rates, and reproduction numbers [6]. Although often associated with viruses, phylodynamics also applies to macroevolution, immune-cell lineages, and cancer evolution [4]. In this thesis, however, the focus is specifically on viral phylodynamics.

A major class of phylodynamic models is based on **birth–death models**, where lineages "birth" by transmitting and "die" by becoming noninfectious. Birth–death models provide a flexible mathematical foundation for studying epidemic dynamics and have been widely used across evolutionary biology [7].

1.3 The challenge: Detecting evolutionary communities

Real phylogenies often contain substructures, clades or lineages, that correspond to different evolutionary or epidemiological behaviors. Detecting these substructures is important for identifying new variants, understanding shifts in fitness, and analyzing patterns of transmission.

This task resembles **community detection** in network science, where the goal is to find groups of nodes that are more similar to each other than to the rest of the graph. However, detecting meaningful communities in phylogenetic trees remains a difficult task. Most current approaches still rely on manual definitions of clades or lineage labels created by expert committees (such as Pango for SARS-CoV-2), which often depend on heuristic rules and do not scale well to large datasets [8], with one of the most recent attempts being Phylowave [9], which relied on statistical methods for clade detection.

Importantly, although deep learning has been used in phylogenetics for parameter inference and simulation-based modeling [10, 11], recursive neural networks have not yet been explored as tools for clade or community detection in phylogenies. This leaves an open opportunity for data-driven methods that can automatically learn evolutionary structure, rather than relying on hand-crafted and pathogen-specific rules.

[1] a monophyletic group is a set of organisms that includes one common ancestor and all of its descendants.

1.4 Existing solutions: Deep learning as a surrogate model

Deep learning has increasingly been used as a surrogate for mathematically complex or computationally expensive evolutionary models. For example, neural networks have been applied to infer diversification and birth–death parameters directly from phylogenies [11]. Other recent work shows that it is possible to learn pathogen fitness from phylogenies using neural networks, producing estimates that match or exceed classical model-based approaches [9]. These studies demonstrate that deep learning can extract meaningful evolutionary information from tree topology and branch lengths without requiring explicit likelihood formulas.

At the same time, classical phylogenetics provides important tools for understanding the limits of tree-based inference. Foundational work on phylogenetic reconstruction introduced many of the methods still used today [5], and later developments in molecular-clock modeling showed how to estimate evolutionary timescales from genetic data [12]. Birth–death skyline approaches provide fast and nonparametric estimates of population size dynamics [13]. These results highlight that while strong mathematical tools exist, they can become difficult to apply or scale when phylogenies grow large.

Together, these findings show both (i) the promise of machine learning for scalable inference and (ii) the need for new methods specifically aimed at detecting clades or evolutionary communities.

1.5 Proposed work: A deep-learning framework for clade detection

This thesis introduces **Phyloscope**, a new method for detecting evolutionary communities in phylogenetic trees. The framework is based on three main components:

- **recursive neural encodings** inspired by early work on processing structured data such as trees and graphs [14];
- a **contrastive learning** objective that leads samples from the same clade to have similar embeddings [15];
- a simulation pipeline generating **mutated phylogenies** under birth–death models, which provide trees with known fitness shifts for supervised training [7].

Phyloscope learns an embedding for each node in the tree, and clades can be identified by applying a similarity threshold in this latent space. This design aims

to provide a scalable and flexible alternative to heuristic lineage definitions, with the ability to generalize across pathogens and evolutionary scenarios.

1.6 Thesis structure

This thesis is organized as follows:

- **Chapter 2 — Background**: introduces birth–death models, recursive neural networks, and existing approaches to clade detection.
- **Chapter 3 — Materials & Methods**: describes the simulation protocol, model architecture, training pipeline, and empirical datasets.
- **Chapter 4 — Results**: presents validation on simulated phylogenies and application to real pathogen data.
- **Chapter 5 — Discussion & Conclusions**: summarizes the contributions, discusses limitations, and outlines future research directions.

Chapter 2

Background

2.1 Birth-Death models

Birth-Death Models are continuous-time Markov chains used to study model population sizes through time [16].

In birth-death models two possible events can occur:

- **birth**: the number of lineages increases by one
- **death**: the number of lineages decreases by one

These two events are described by two different rates:

- λ: birth rate
- ψ: death rate[1]

Given that during birth, the number of individuals is increased only by one, an important property of these models is that there are no hard polytomies.[2]

These two events follow a Poisson Process, this means that the expected waiting times for the next event have an exponential distribution with parameter $(\lambda + \psi)$. If there are $N(t)$ lineages alive at time t, then the waiting time for the next event is an exponential distribution with parameter $N(t)(\lambda + \psi)$.

Probabilities of birth and death can be derived in a short time interval Δt, where at most one event can occur, as:

[1] sometimes written as γ

[2] A polytomy is a section of the phylogenetic tree where a single lineage splits into three or more descendant lineages simultaneously.

$$P(\text{birth}) \approx N(t)\lambda\Delta t \tag{2.1}$$
$$P(\text{death}) \approx N(t)\psi\Delta t \tag{2.2}$$

The expected value of $N(t)$ after a short time interval Δt is:

$$N(t+\Delta t) = N(t) + N(t)\lambda\Delta t - N(t)\psi\Delta t \tag{2.3}$$

Converting this expression into a differential equation by subtracting $N(t)$ from both sides, then dividing by Δt and taking the limit as Δt becomes very small:

$$\frac{dN}{dt} = N(\lambda - \psi) \tag{2.4}$$

Solving the differential equation, integrating both sides and solving them, defining the constant $N(0) = n_0$, which means that at time 0 n_0 lineages are present:

$$N(t) = n_0 e^{(\lambda-\psi)t} \tag{2.5}$$

Fig.2.1 shows three different behaviors with variation in the quantity $\lambda - \psi$ when $n_0 = 1000$

Denoting as $p_n(t) = P[N(t) = n]$ for $n \geq 0$ the full probability of the model:

$$p_0(t+\Delta t) = p_1(t)\psi\Delta t + p_0(t) \tag{2.6}$$
$$p_1(t+\Delta t) = p_1(t)(1 - (\lambda+\psi)\Delta t) + p_2(t)\cdot 2\psi\Delta t \tag{2.7}$$

This is the probability of having zero lineages at time $t + \Delta t$. It is given by the probability that there is one lineage at time t and a death event occurs during Δt, plus the probability that there are already zero lineages at time t and no event occurs.[3]

The probability of having one lineage alive at time $t + \Delta t$ corresponds to the probability that the population still consists of a single lineage at time t, after accounting for the possible birth and death events during the interval Δt, together with the probability that, starting from two lineages at time t, exactly one death event occurs.[4]

By induction, also accounting for birth events:

[3]With zero lineages, neither birth nor death events can occur.

[4]A birth event cannot occur when starting from a single lineage and still result in one lineage at time $t + \Delta t$.

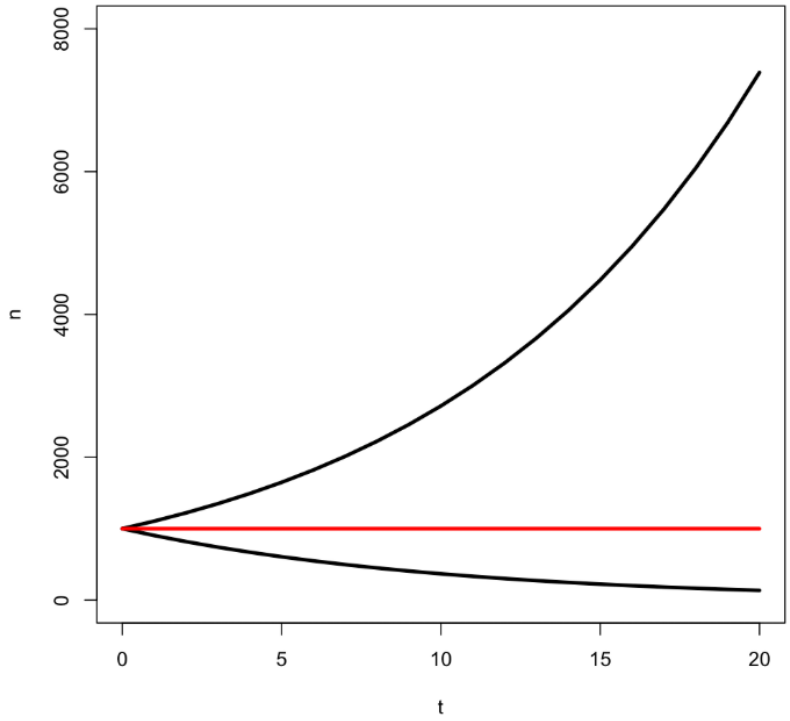


Figure 2.1: Expected number of species under a birth-death with the number of lineages $n = 1000$: **top line** $\lambda - \psi > 0$, **middle line** $\lambda - \psi = 0$, **bottom line** $\lambda - \psi < 0$. Figure taken from [16]

$$p_n(t+\Delta t) = p_{n-1}(t)(n-1)\lambda\Delta t + p_{n+1}(t)(n+1)\psi\Delta t + p_n(t)(1-n(\lambda+\psi)\Delta t) \quad (2.8)$$

Subtracting $p_n(t)$ from both sides, dividing both sides for Δt and taking the limit for Δt that becomes smaller, 2.8 can be converted into a set of differential equations:

$$\frac{dp_n(t)}{dt} = p_{n-1}(t)(n-1)\lambda + p_{n+1}(t)(n+1)\psi - p_n(t)n(\lambda+\psi) \quad (2.9)$$

Equation 2.9 is the continuous-time master equation governing the full Birth–Death process. It characterizes the temporal evolution of the probability distribution $\{p_n(t)\}_{n\geq 0}$ over the number of lineages. Although solving this infinite system explicitly is rarely feasible, the equation provides the formal probabilistic foundation for birth–death dynamics and serves as the starting point for deriving analytical results, approximations, and likelihood functions used in phylodynamic inference.

This concludes the derivation of the Birth–Death process in discrete and continuous form.

2.1.1 Phylogenetic Trees

Keeping track of the parent-offspring relationships among lineages, the phylogenetic tree can be obtained. This represents the pathogen history, where each branching in the tree represents a transmission event (e.g. a birth).

In Fig. 2.2, the three configurations highlight how different observation and sampling schemes shape the appearance of the underlying evolutionary process. The first tree reflects the complete history, retaining every branching event. The second shows how removing extinct lineages produces a more compact representation of the survivors. The third illustrates the additional incompleteness introduced by partial sampling, where only a subset of existing lineages is captured, altering both the depth and overall structure of the resulting phylogeny. The last representation is the partial reconstruction of the pathogen through sampling. Since the complete transmission history cannot be observed, only the reconstructed tree is available.

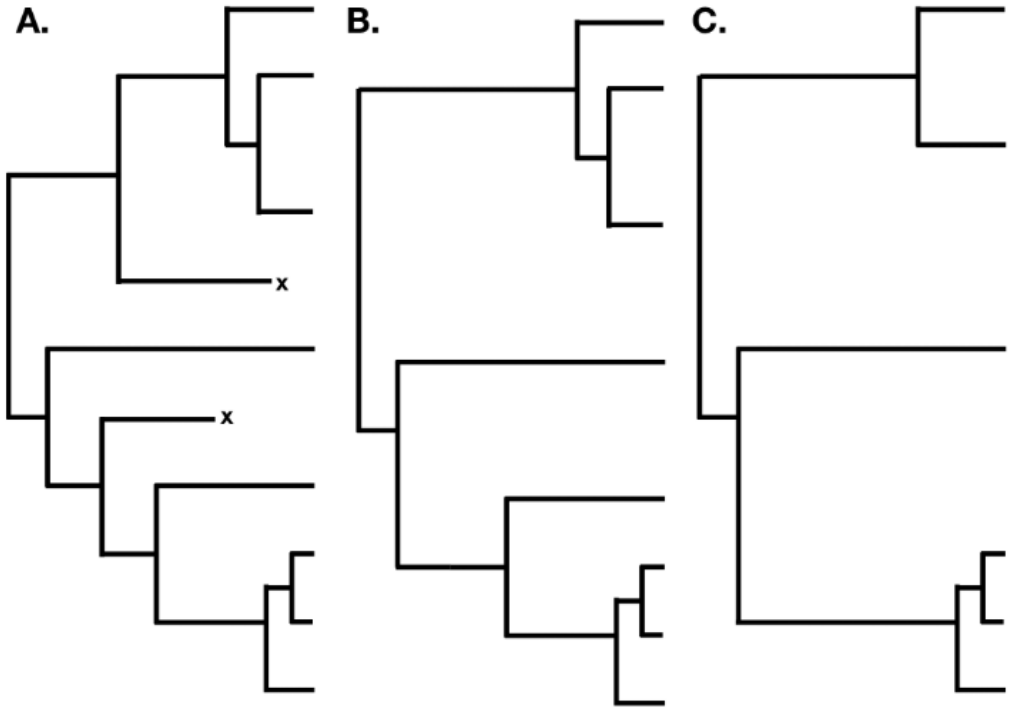


Figure 2.2: Examples of three different phylogenetic trees. **A**: the true phylogenetic tree, the tree obtained observing each lineage evolving over time. **B**: the extant-species tree, the same tree considering only the lineages still alive today. **C**: a partially sampled phylogenetic tree. A subset of lineages is sampled. Figure taken from [16]

Tree characteristics

The following are three relevant characteristics of phylogenetic trees:

- **Tree topology**: independent of the branch lengths of the phylogenetic tree.
- **Tree shape**: ignores both branch lengths and tip labels.
- **Tree balance**: identifies the differences in the number of descendants at different points of the tree.

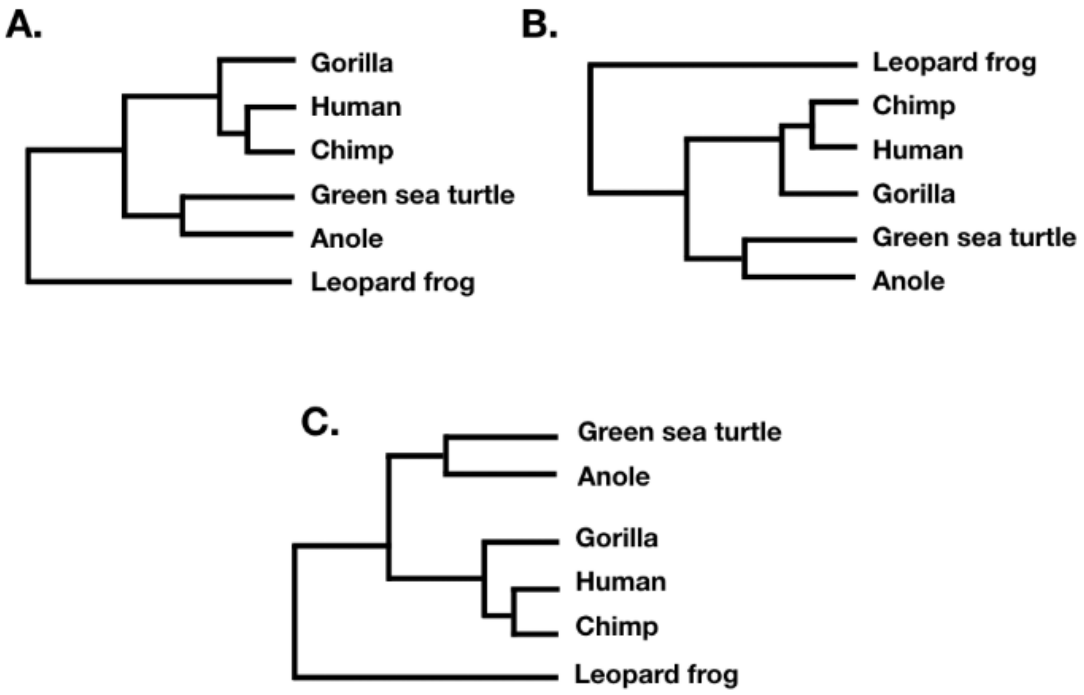


Figure 2.3: Trees sharing the same tree topology. The clades and branching order remain unchanged, even though the drawings differ in orientation and branch lengths. Figure taken from [16]

In Fig.2.3 the three phylogenetic trees (A–C) display the same topology, as they represent identical evolutionary relationships among the taxa. Differences in the graphical layout do not affect the tree structure. In all trees, the clade formed by Gorilla, Human, and Chimp remains consistent, as does the clade including Green sea turtle and Anole, with the Leopard frog branching off earlier as the outgroup. The only variation that may occur between equivalent topologies

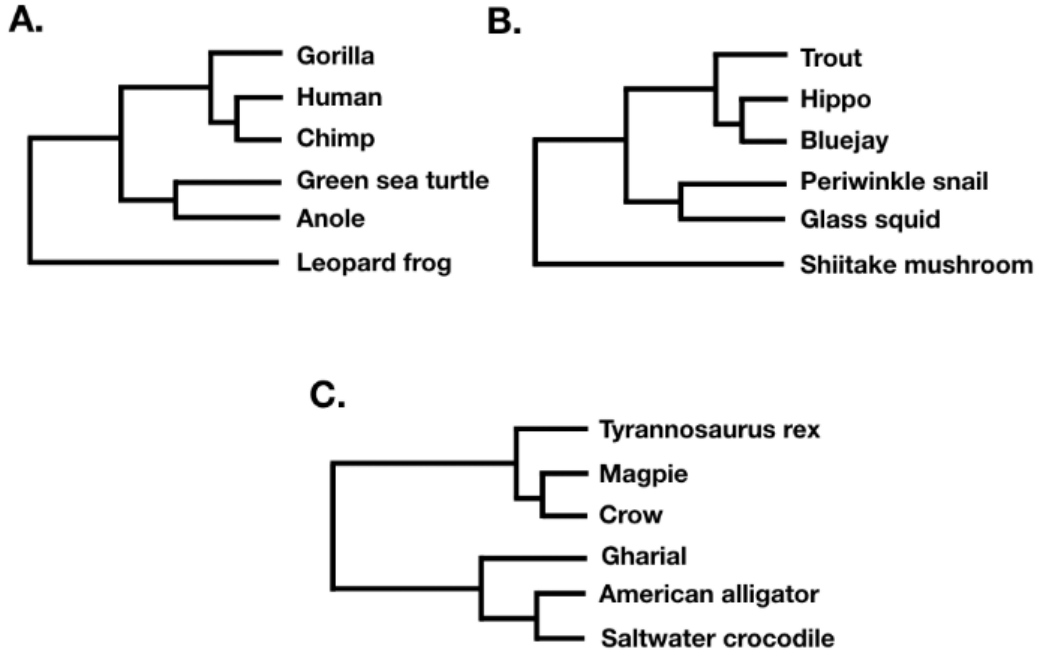


Figure 2.4: Examples of phylogenetic trees that share the same overall tree shape (A and B) and one with a different shape (C). Trees A and B display an identical branching structure once tip labels are ignored: both exhibit an unbalanced configuration in which most lineages belong to one major clade. Tree C instead shows a more balanced division of lineages, illustrating how tree shape captures the distribution of descendants across major splits independently of the specific taxa involved. Figure taken from [16]

concerns the branch lengths, which indicate the amount of evolutionary change or time separating the nodes.

In Fig.2.4, considering the deepest split in the tree, A and B present an unbalanced structure $(5, 1)$, while C presents a balanced structure $(3, 3)$.

Starting from a single node n in the phylogenetic tree, there are two clades descending from this node, a and b. The total number of species descending from that node is $N_{\text{total}} = N_a + N_b$ with $N_a, N_b \neq 0$.[5] All possible divisions between N_a and N_b are equally probable (e.g, (9,1), (8,2) etc.)[6], so that each can occur with probability 1/9. Generalizing, we have:

[5] N here refers to the number of species, not the number of lineages

[6] if the clades are unlabeled then (6,4) and (4,6) coincide.

$$P(N_a \mid N_{\text{total}}) = \frac{1}{N_{\text{total}} - 1} \tag{2.10}$$

Tree balance statistics provide a way to check how the species in the tree are distributed. The simplest and most used index is Colless' index [17], denoted I_C:

$$I_C = \frac{\sum |N_L - N_R|}{\frac{(N-1)(N-2)}{2}} \tag{2.11}$$

where N_L and N_R are the number of species on the left and right side of the tree, respectively. If the tree is pectinate (heavily unbalanced), this index will be equal to 1. On the contrary, if the tree is perfectly balanced, Ic will be equal to zero.

Another important index that quantifies how balanced a tree is, is the Sackin index [17], denoted I_S. This index measures the overall depth of the tips (leaves) in a phylogenetic tree. Formally, it is defined as the sum of the depths of all terminal nodes, where the depth of a tip corresponds to the number of internal nodes between that tip and the root of the tree:

$$I_S = \sum_{i=1}^{N} d_i \tag{2.12}$$

where d_i is the depth of tip i, and N is the total number of tips (species) in the tree. Intuitively, I_S increases as the tree becomes more unbalanced, since in unbalanced trees many leaves are located at greater depths. In contrast, perfectly balanced trees minimize this sum, having all tips at equal distances from the root.

To allow comparisons among trees with different numbers of tips, a normalized I_S can be computed by dividing it by its maximum or expected value under a null model.

Thus, in general:

- a high I_S indicates a strongly unbalanced (pectinate) tree;
- a low I_S index indicates a more balanced tree, where most lineages diversify at similar rates.

A set of contrasting tree structures is shown in Fig. 2.5 to illustrate how different branching patterns reflect varying degrees of balance in a phylogeny. The caterpillar tree represents an extreme case of asymmetry, in which diversification proceeds almost linearly, producing long chains of single splits. In contrast, the fully symmetric bifurcating tree distributes lineages evenly at each branching event, resulting in the most balanced configuration. The star tree exemplifies the opposite extreme: all lineages originate from a single ancestral split, collapsing much of

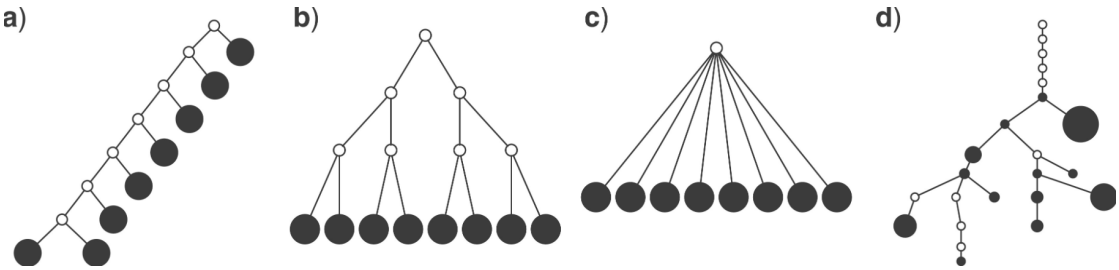


Figure 2.5: Example of four different tree structures. **a**: caterpillar tree, the most unbalanced tree, with $I_S = 35$, $I_S, norm = 1$ $I_C = 21$, $I_C, norm = 1$. **b**: fully symmetric bifurcating tree, with $I_S = 24$, $I_S, norm = 0.59$ $I_C = I_C, norm = 0$ **c**: star tree, with $I_S = 8$, $I_S, norm = 0$ **d**: clone tree of the lung tumor CRUK0065 in the TRACERx cohort. Nodes represented by empty circles correspond to extinct clones, and the diameters of other nodes are proportional to the corresponding clone population sizes. Figure taken from [17]

the hierarchical structure of the tree. Finally, the empirical clone tree highlights how real biological systems can display complex combinations of balanced and unbalanced regions, as well as extinct lineages and uneven clone sizes. Together, these examples show how tree shape captures important aspects of diversification dynamics and the historical processes generating observed phylogenies.

Lineage-through-time Plots

Lineage-through-time (LTT) plots describe how the number of lineages changes over time. An example of LTT plot is shown in Figure 2.6. Time is represented on the x-axis and the reconstructed number of lineages on the y-axis. Since given that the population size typically increases exponentially through branching, it is common practice to log-transform the y-axis to better visualize growth patterns.

Figure 2.6 illustrates the direct correspondence between a phylogenetic tree (top) and its lineage-through-time representation (bottom). Each *red dashed line* indicates the time at which an internal node occurs in the tree, that is, a birth event or transmission event in the underlying birth-death process. At each of these points, the LTT curve displays an upward step, reflecting an increase in the number of extant lineages.

Between two successive branching events, the LTT curve remains flat, indicating that no new lineages have been generated in that interval. The overall slope of the curve therefore provides an intuitive summary of the diversification dynamics: a *steeper* slope corresponds to a period of rapid branching (high transmission rate), while a *shallower* slope indicates slower diversification.

Because the LTT plot is constructed directly from the internal nodes of the tree,

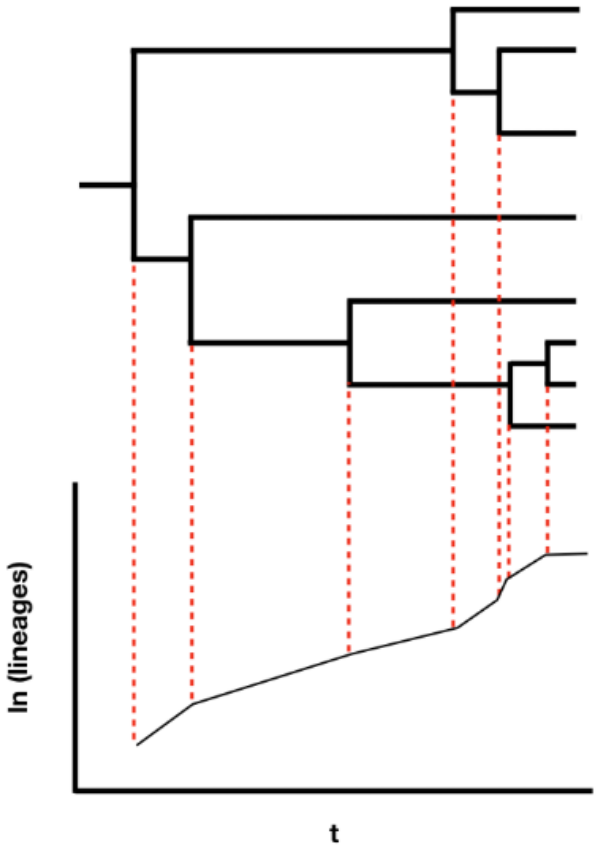


Figure 2.6: Example of LTT plot: red lines correspond to transmission events and are linked from the phylogenetic tree to the LTT plot. Figure taken from [16]

it acts as a compact visualization of the tempo of the evolutionary or epidemiological process.

2.1.2 Multi-type birth-death models

The simple Birth-Death (BD) model is the simplest example of the Multi-Type Birth-Death Model (MTBD) family [18]. The MTBD is a birth-death process in which the individuals have a type label or state label (m states) which categorizes their properties. As we explain in detail below, these types may correspond to the individuals having different properties, for example, a higher or lower birth/death rate, or a specific stage of infection. This multi-state approach allows the model to capture heterogeneity and structural complexity in a population that the simple BD model cannot.

MTBD parameters An MTBD model with m states has $m(m-1)$ transition rate parameters, m^2 transmission rate parameters, m removal rates[7] and m sampling probabilities upon removal parameters.

- μ_{ij}: the transition rate parameter indicates the rate with which from state i we go to state j, where $i \neq j$.
- λ_{ij}: the transmission rate parameters indicates the rate with which i go from from state i (donor) to state j (recipient)[8].
- ψ_i: the removal rate of state i.
- p_i: probability to sample the pathogen of an individual in state i upon removal[9].

An MTBD has the following epidemiological parameters:

- Reproduction number of state i:

$$R_i = \frac{\sum_{1 \leq j \leq m} \lambda_{ij}}{\psi_i} \tag{2.13}$$

- Exit time[10] from state i:

$$d_i = \frac{1}{\sum_{\substack{1 \leq j \leq m \\ j \neq i}} \mu_{ij} + \psi_i} \tag{2.14}$$

Example models Fig.2.7 shows the most famous types of birth-death models with their respective formulas.[11]

The last variant, the birth-death model with Exposed-Infectious and Super-Spreading individuals (BDEISS) model, not shown in the Figure, is only occasionally used in practice, since it is by far the most complex to handle. Its structure involves several interacting states and parameters, which significantly complicates both the mathematical analysis and the practical implementation of the model.

[7] Previously the transmission rate was called birth rate and the death rate was called extinction rate, respectively: the names are equivalent

[8] There is a transmission state even when $i = j$

[9] Sampling probability will sometimes be called s_i.

[10] The reader is advised to distinguish between parameters ψ and μ, ψ is the rate at which the state is removed by the model, this means it ceases to exist in the model. μ is the transition rate from one state to another, still being active in the whole model.

[11] In the figure β represents the birth rate and γ the death rate

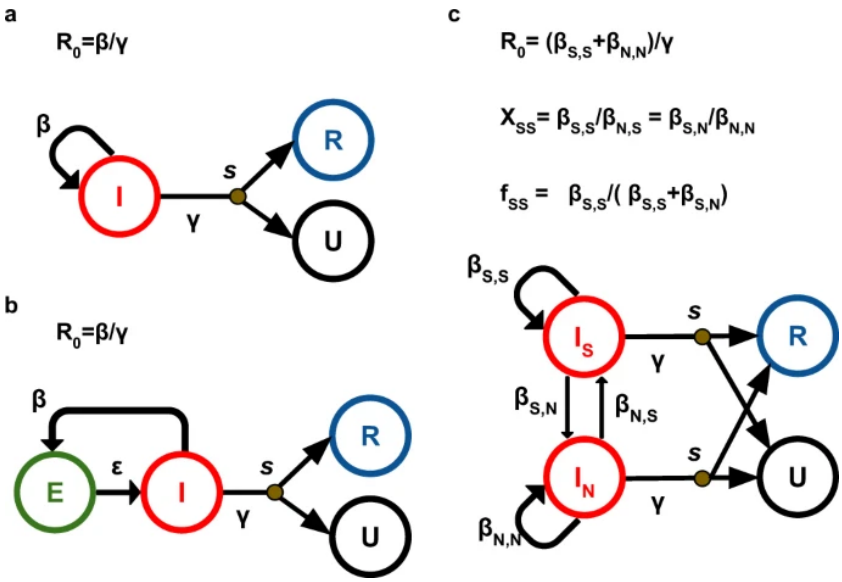


Figure 2.7: **a** birth-death model (BD), **b** birth-death model with Exposed-Infectious individuals (BDEI), **c** birth-death model with SuperSpreading (BDSS) [10]

Table 2.1: Overview of Birth-Death Models (BD, BDEI, BDSS)

Feature	**BD**	**BDEI**	**BDSS**
States	1 I	2 E, I	2 I, S
Total Parameters	3	4	5
Rates	$\lambda = \lambda_I$ $\psi = \psi_I$ $p = p_I$	$\mu = \mu_{EI}$ $\lambda = \lambda_{IE}$ $\psi = \psi_I$ $p = p_I$	$\lambda_{II} = \lambda_{nn}$ $\lambda_{IS} = \lambda_{ns}$ $\lambda_{SI} = \lambda_{sn}$ $\lambda_{SS} = \lambda_{ss}$ $\psi_I = \psi_S$ $p_I = p_S$
Epidemiological Parameters	$R = \frac{\lambda}{\psi}$ $d_I = \frac{1}{\psi}$	$d_E = \frac{1}{\mu}$	$X_S = \frac{\lambda_{ss}}{\lambda_{ns}} = \frac{\lambda_{sn}}{\lambda_{nn}}$ $f_S = \frac{\lambda_{ss}}{\lambda_{sn} + \lambda_{ss}}$

Table 2.1 summarizes the parameterization and key epidemiological outputs for three primary single-type and multi-type Birth-Death (BD) models. Note that rates (λ, ψ, μ) are instantaneous rates, while p is a probability.

The BDSS (Birth-Death SuperSpreading) model is designed to capture heterogeneity in transmission efficiency by distinguishing between Normal Spreaders (I) and Superspreaders (S).

- **Transition Rates (λ_{XY}):** Since there are two types, we need four transmission rates: λ_{XY} is the rate at which an individual of type **X** gives birth to an individual of type **Y**.
  - λ_{II} (λ_{nn}) and λ_{SS} (λ_{ss}) represent transmission within the same type.
  - λ_{IS} (λ_{ns}) and λ_{SI} (λ_{sn}) represent transmission across types.
- **Removal and Sampling Rates:** The parameters ψ (removal rate) and p (sampling probability) are constrained to be equal across both types ($\psi_I = \psi_S$ and $p_I = p_S$). This makes the sampling independent of the individual's spreading type.
- **Key Epidemiological Parameters:**
  - $\mathbf{X_S}$ (Super-spreading transmission ratio) quantifies the relative infectiousness of the Superspreader type compared to the Normal Spreader type.
  - $\mathbf{f_S}$ (Super-spreading fraction) measures the probability of a transmission event leading to a Superspreader state, given the current infection structure.

2.1.3 Community Detection problem

In the context of evolutionary genomics, the community detection problem can be reformulated as the task of identifying groups of pathogen lineages that share similar evolutionary trajectories within a phylogenetic tree. Traditionally, community detection refers to the process of finding densely connected subgraphs within a larger network, where nodes are more strongly associated with others in the same community than with nodes outside it [19, 20].

In a phylogenetic context, clades correspond to monophyletic groups: subsets of the tree that include an ancestral lineage and all of its descendants. As illustrated in Fig. 2.8, each colored region highlights one such clade, showing how a single branching event gives rise to a group of lineages sharing a common origin. Because members of the same clade emerge from the same ancestral node, they often exhibit similar diversification patterns, mutation profiles, or epidemiological behaviors. For this reason, clades naturally represent "communities" within the tree, providing biologically meaningful groupings that arise directly from the hierarchical structure of the evolutionary process.

When applied to phylogenetic data, this notion translates naturally to the identification of clades, subtrees of closely related genomes that may exhibit

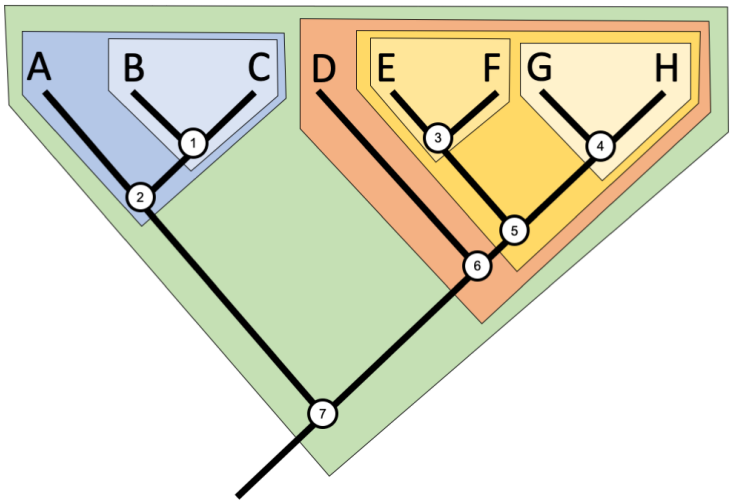


Figure 2.8: The three colored groups (green, red, and blue) represent three monophyletic clades. Each clade corresponds to a subtree defined by a single ancestral node and all of its descendants, highlighting distinct evolutionary lineages within the phylogeny. Image adapted from the Digital Atlas of Ancient Life (CC-BY 4.0).

coherent patterns of diversification, mutation accumulation, or epidemiological behavior.

Each phylogenetic tree represents a historical record of pathogen evolution, where nodes correspond to sampled genomes and edges represent ancestral relationships derived from genetic divergence [21, 8]. Communities within such trees emerge as topologically and dynamically distinct regions, corresponding to evolutionary subpopulations that may differ in fitness or transmission potential. Detecting these substructures is crucial for understanding how new variants arise, expand, and potentially outcompete other circulating lineages [2].

However, unlike standard networks, phylogenetic trees have a strict hierarchical and acyclic structure that constrains the topology of possible connections. This means that community detection in a phylogenetic context cannot rely on traditional modularity-based or stochastic block model approaches [19]. Instead, it requires methods that can capture both the tree topology and the temporal or mutational dynamics encoded within it. Conventional approaches such as clade-based heuristics or threshold clustering often rely on manually defined criteria, (e.g., fixed genetic distance cutoffs or clade-size thresholds), which are sensitive to dataset-specific biases and lack generalization across pathogens [3].

Recent advances in data-driven modeling offer a new perspective on this problem.

By framing the phylogenetic tree as a structured graph, graph-based neural architectures—including recursive networks and message-passing neural networks—can learn to embed nodes (genomes or lineages) into a latent representation space [14]. In this space, communities correspond to clusters of nodes with similar evolutionary signals, such as shared fitness dynamics or mutation profiles [9]. Contrastive learning provides a particularly effective training paradigm for this purpose: it encourages the model to bring together embeddings of nodes that belong to the same evolutionary regime while pushing apart those that follow different dynamics [15].

Through this lens, community detection becomes a representation-learning problem rather than a purely structural one. The goal is not only to partition the phylogeny, but to learn a latent space where evolutionary similarity is measured continuously. This allows the model to generalize across diverse phylogenetic configurations and capture subtle shifts in fitness that may not be visible through manual analysis or static clustering thresholds [10, 11].

Ultimately, community detection in phylogenetic trees bridges the gap between computational graph theory and evolutionary biology. It provides a scalable framework for detecting emerging variants, interpreting evolutionary pressures, and quantifying lineage-level differences in transmissibility or adaptation [2]. By replacing heuristic classification with learned representations, this paradigm paves the way toward a fully data-driven, automated form of variant discovery, a central goal of modern genomic surveillance [8].

2.2 Neural Networks for Tree Data

This work focuses exclusively on phylogenetic trees with *temporal* branch lengths. These time-scaled (or dated) trees are the standard objects of inference in phylodynamics, as their branch lengths directly represent evolutionary time and encode the tempo of the epidemic process [21]. A dated tree $\mathcal{T}$ therefore contains both the topology (the ancestor–descendant relationships) and the temporal structure of the outbreak, providing essential information for reconstructing transmission dynamics.

The main objective of phylodynamic inference is to infer the epidemiological parameters $\boldsymbol{\theta}$ that generated the observed phylogeny. These include, for instance, the basic reproduction number R_0, the infectious period $1/\psi$, the latency period $1/\epsilon$, or superspreading-related quantities such as f_{SS} or X_{SS}. The problem can be framed as a **maximum-likelihood estimation** (MLE) task:

$$\hat{\boldsymbol{\theta}}_{\mathrm{MLE}} = \arg\max_{\boldsymbol{\theta}} P(\mathcal{T} \mid \boldsymbol{\theta}), \tag{2.15}$$

where $P(\mathcal{T} \mid \boldsymbol{\theta})$ is the probability of observing the dated tree $\mathcal{T}$ under a given

birth–death model.

For simple models such as the basic Birth–Death (BD) process, closed-form likelihood expressions allow efficient inference. However, more expressive models, such as the Birth–Death with Exposed and Infectious stages (BDEI) or the Birth–Death with Superspreading (BDSS), require solving systems of differential equations, making likelihood evaluation computationally demanding [10]. State-of-the-art Bayesian tools such as `BEAST2` or maximum-likelihood approaches such as `TreePar` depend on numerical approximations whose computational cost grows rapidly with tree size or model complexity [10, 21].

To overcome these limitations, amortized Bayesian approaches [22] use neural networks to approximate the mapping between a tree and its generating parameters. We define a neural estimator

$$\hat{\boldsymbol{\theta}} = f_{\boldsymbol{\phi}}(\mathcal{T}), \tag{2.16}$$

where $f_{\boldsymbol{\phi}}$ is a neural network parameterized by $\boldsymbol{\phi}$, trained to predict the parameters associated with an input dated tree. During training, the model minimizes the expected error over a large collection of simulated phylogenies:

$$\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{(\mathcal{T},\boldsymbol{\theta})\sim\mathcal{D}}\left[\left\|f_{\boldsymbol{\phi}}(\mathcal{T}) - \boldsymbol{\theta}\right\|^2\right], \tag{2.17}$$

with $\mathcal{D}$ denoting the joint distribution of simulated trees and known ground-truth parameters.

Once trained, the neural estimator becomes a **likelihood-free surrogate model**: it provides near-instantaneous predictions of $\boldsymbol{\theta}$ for any new observed tree, bypassing differential equations and Monte Carlo Markov Chains (MCMC) sampling entirely. This paradigm, known as amortized or simulation, based inference, is increasingly used in phylodynamics and infectious disease modeling [10].

In the remainder of this chapter, we discuss strategies for encoding dated phylogenetic trees into neural-compatible formats and review the main architectures proposed for this task, ranging from summary statistics and compact encodings to graph-based and recursive neural models [14, 11, 22].

2.2.1 How to use Neural Networks for Surrogate Deep Learning

Traditional phylodynamic inference relies on explicit likelihood-based methods such as `BEAST2` or `TreePar` [10]. These frameworks estimate epidemiological parameters, like the basic reproduction number (R_0), infectious period ($1/\gamma$), or latency period ($1/\psi$), by maximizing the likelihood of observing a given phylogenetic tree under a specified birth-death model. However, as model complexity increases (e.g., with latent states or superspreading), the associated differential equations become computationally intractable [21].

Surrogate deep learning offers an efficient alternative by training neural networks to approximate the mapping between phylogenetic trees and model parameters. Once trained, these models perform inference orders of magnitude faster than likelihood-based approaches while maintaining comparable accuracy. In this framework, neural networks act as learned estimators of epidemiological parameters, trained on large numbers of simulated trees generated under known parameters [22].

This paradigm was pioneered by *PhyloDeep*, where millions of simulated trees under BD, BDEI, and BDSS models were used to train feed-forward and convolutional architectures [10]. Once the network learns to infer parameters from synthetic trees, it can generalize to real outbreak data, bypassing explicit likelihood computation. Similar ideas have been investigated in broader diversification contexts, further confirming the importance of choosing appropriate neural architectures and tree representations for accurate parameter recovery [11].

Such surrogate models provide a scalable approach for epidemic inference, enabling near real-time phylodynamic analysis and complementing classical Bayesian or maximum-likelihood frameworks [8].

2.2.2 Summary statistic representations

One strategy to make phylogenetic trees compatible with neural networks is to represent them through summary statistics (SS). These are handcrafted numerical features describing the topology and branch lengths of the tree, for example, lineage-through-time (LTT) slopes, tree balance indices, and distributions of coalescent intervals [10].

In PhyloDeep [10], a set of 83 summary statistics was used for BD and BDEI models, extended to 97 for BDSS to capture superspreading effects. Each tree is thus transformed into a fixed-length vector that feeds a Feed-Forward Neural Network (FFNN), trained to regress model parameters and classify the underlying birth–death model. This approach offers simplicity and interpretability. The more statistics are included, the more precise and faithful the representation is, the loss of information with respect to the phylogenetic tree is inevitable [10].

2.2.3 Compact Bijective Ladderized Vector (CBLV) representation

To address the limitations of the summary statistics framework, an alternative approach was invented in PhyloDeep. The CBLV [10] encoding provides a compact, bijective transformation of the phylogenetic tree into a vector suitable for deep learning. The process involves two key steps:

1. **Ladderization**: for each internal node, the subtree containing the most recently sampled tip is rotated to the left, standardizing the orientation of the tree.

2. **Inorder Traversal**: during traversal, for internal nodes the distance from the root is appended to a real-valued vector, and for each visited tip its distance from the previously visited internal node.

The resulting vector is padded with zeros for the maximum tree size. This vectorized representation preserves both tree topology and branch length information, making it suitable as direct input to Convolutional Neural Networks (CNNs). In the PhyloDeep framework, CNNs trained on CBLV-encoded trees achieved accuracy comparable to FFNNs trained on summary statistics, while offering better generalization across different epidemic models. The main tradeoff lies in higher computational cost during training but increased flexibility and model reuse for unseen tree structures.

Fig. 2.9 shows PhyloDeep pipeline, from CBLV and summary statistics representations to the neural networks used in both scenarios. Finally, their deployment for inference of epidemiological parameters and model selection.

2.2.4 Graph neural network approaches

While CBLV encodings allow vector-based learning, they lose some of the relational structure inherent to phylogenetic trees, especially because the CNN has a limited receptive field. Graph Neural Networks (GNNs) overcome this limitation by operating directly on the tree as a graph, where nodes represent sequences or ancestors and edges encode evolutionary relationships. This makes them well-suited for learning on relational data structures more complex than sequences or grids, as highlighted in broader surveys on deep learning for networked data [19, 20].

In recent work comparing multiple neural architectures, (FFNNs, CNNs, RNNs, and GNNs), GNNs have demonstrated the ability to exploit tree topology through message-passing mechanisms. This enables local feature aggregation and hierarchical representation learning across the entire phylogeny without flattening it into Euclidean space, a result also supported in comparative studies on deep learning for diversification and phylogenetic inference [11]. Although the GNN implementations deployed suffered from over-smoothing[12].

A recent and notable example is *DeepDynaForecast*, which introduced a Primal-Dual Graph LSTM (PDGLSTM) to jointly model nodes and edges of phylogenetic

[12]loss of node distinction after multiple layers.

trees [23]. By learning temporal and structural dependencies, the model predicts short-term transmission dynamics for each sample in an outbreak. When applied to large-scale HIV and SARS-CoV-2 datasets, DeepDynaForecast substantially outperformed standard GNN and GCN baselines, demonstrating the potential of graph-based neural models for real-world genomic surveillance.

2.2.5 Towards recursive architectures

Recursive neural networks (RvNNs) provide a mathematically principled framework for processing trees of variable size. Unlike CNNs or GNNs, an RvNN defines a function f over trees recursively:

$$f(\mathcal{T}_i) = \begin{cases} \sigma(\mathbf{W}_\ell \mathbf{x}_i + \mathbf{b}_\ell), & \text{if } i \text{ is a leaf,} \\ g_\theta\Big(f(\mathcal{T}_{\text{LEFT}(i)}), f(\mathcal{T}_{\text{RIGHT}(i)}), \mathbf{x}_i\Big), & \text{otherwise.} \end{cases} \tag{2.18}$$

Here $f(\mathcal{T}_i) \in \mathbb{R}^m$ is the embedding of the subtree rooted at i, and g_θ is typically a feed-forward transformation such as

$$g_\theta(\mathbf{h}_L, \mathbf{h}_R, \mathbf{x}_i) = \sigma\Big(\mathbf{W}\,[\mathbf{h}_L \,\|\, \mathbf{h}_R \,\|\, \mathbf{x}_i] + \mathbf{b}\Big), \tag{2.19}$$

where $\|$ denotes concatenation. This formulation naturally mirrors the process of tree construction: information flows bottom-up from tips to root, aggregating progressively larger evolutionary contexts. The recursive computation can be efficiently implemented via dynamic programming and differentiated using backpropagation through structure (BPTS), as formalized in the original framework for recursive networks [14].

RvNNs therefore preserve both the hierarchical and chronological structure of the data, avoid arbitrary traversal orders, and support variable-size input without padding. When combined with contrastive or supervised objectives, they enable the model to learn embeddings that reflect shared evolutionary dynamics, precisely the type of structure required to identify clades undergoing distinct fitness changes, as highlighted in recent work on learning fitness dynamics from phylogenies [9].

2.3 Contrastive Learning

Contrastive learning has recently emerged as a powerful paradigm in machine learning, particularly for representation learning in complex or structured domains. Its key principle is to learn an embedding space in which samples that share similar semantic or structural properties are mapped close to each other, while dissimilar samples are placed farther apart. This approach is especially suitable for

phylogenetic applications, where evolutionary similarity can be naturally interpreted as proximity in an abstract latent space [21].

2.3.1 Foundations of Contrastive Learning

Let $\mathcal{X}$ denote the input space (e.g., a set of phylogenetic trees or nodes within a tree), and let $f_{\boldsymbol{\phi}} : \mathcal{X} \to \mathbb{R}^d$ be an encoder parameterized by $\boldsymbol{\phi}$ that maps each input $x \in \mathcal{X}$ to a d-dimensional representation vector $z = f_{\boldsymbol{\phi}}(x)$. The central idea of contrastive learning is to train $f_{\boldsymbol{\phi}}$ so that representations of "positive" pairs are close, while those of "negative" pairs are far apart.

Given a dataset of paired examples $\{(x_i, x_i^+, x_i^-)\}$, where x_i^+ is a positive (similar) example of x_i and x_i^- a negative (dissimilar) one, the learning objective can be formalized as the minimization of a contrastive loss function. A common choice is the **InfoNCE** (Noise Contrastive Estimation) loss [15]:

$$\mathcal{L}_{\text{InfoNCE}} = -\sum_{i=1}^{N} \log \frac{\exp\Big(\text{sim}(z_i, z_i^+)/\tau\Big)}{\exp\Big(\text{sim}(z_i, z_i^+)/\tau\Big) + \sum_{k=1}^{K} \exp\Big(\text{sim}(z_i, z_k^-)/\tau\Big)} \tag{2.20}$$

where $\text{sim}(\cdot, \cdot)$ is a similarity measure, typically the cosine similarity, and $\tau > 0$ is a temperature parameter controlling the sharpness of the distribution.

The minimization of $\mathcal{L}_{\text{InfoNCE}}$ encourages the encoder to produce similar embeddings for related samples and to separate unrelated ones. At convergence, the learned representation space preserves the intrinsic relationships present in the data, even without explicit supervision. This property makes contrastive learning particularly effective in domains such as phylogenetics, where labels describing evolutionary dynamics or lineage membership are often unavailable or incomplete [10].

2.3.2 Contrastive Learning in Hierarchical Data

Contrastive learning can be naturally extended to hierarchical and graph-structured data, such as phylogenetic trees. In this setting, a phylogeny $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ consists of a set of nodes $\mathcal{V}$ (representing sampled sequences or ancestral taxa) connected by edges $\mathcal{E}$ that encode evolutionary relationships. Each node $i \in \mathcal{V}$ can be mapped to a latent representation $z_i = f_{\boldsymbol{\phi}}(i)$ through a neural architecture designed to exploit both the topological and temporal structure of the tree (e.g., recursive neural networks or graph neural networks [19, 20]).

The highlighted clades in Figure 2.10 illustrate groups of taxa that share a recent common ancestor. Such clades provide a natural way to define positive and negative pairs for contrastive learning: samples within the same clade are

considered positive pairs, reflecting their close evolutionary relationship, while samples from different clades serve as negative pairs, representing more distant evolutionary divergence.

A general formulation of the contrastive objective on a phylogenetic tree is:

$$\mathcal{L}_{\text{contrastive}} = -\sum_{(i,j)\in\mathcal{P}} \log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k\in\mathcal{N}(i)} \exp(\text{sim}(z_i, z_k)/\tau)} \tag{2.21}$$

where $\mathcal{P}$ denotes the set of positive pairs and $\mathcal{N}(i)$ denotes all nodes contrasted against i. This loss enforces local smoothness in the embedding space: nodes that are close in evolutionary time or genetic distance acquire similar representations, whereas distant nodes remain well separated.

When applied to graph-structured phylogenies, contrastive learning can be implemented using two complementary encoders:

- a *local encoder*, which aggregates information from immediate neighbors through message passing (as in Graph Neural Networks);
- a *global encoder*, which captures coarse-scale properties of the entire tree, such as its temporal depth or overall diversity.

By contrasting local and global embeddings, the model learns hierarchical representations that capture both fine-scale and macro-evolutionary structure. These representations can subsequently be employed for downstream tasks such as variant detection, lineage clustering, or prediction of evolutionary fitness.

2.3.3 Community Detection as Classification

Once node representations z_i have been learned through contrastive pretraining, community detection in phylogenetic trees can be reformulated as a **classification problem**. The aim is to assign each node i to a community or lineage label $y_i \in \{1, \ldots, K\}$, where K is the number of distinct evolutionary regimes. This can be achieved by adding a classification head g_ψ on top of the encoder:

$$\hat{y}_i = g_\psi(z_i) \tag{2.22}$$

The model is then optimized using a cross-entropy loss:

$$\mathcal{L}_{\text{class}} = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik} \log \hat{y}_{ik} \tag{2.23}$$

In a self-supervised or weakly supervised setting, labels may be replaced by proxy targets derived from the tree structure itself. For example, similarity between a node and its parent can be used as a continuous metric of relatedness:

$$s_i = \text{sim}(z_i, z_{\text{parent}(i)}) \tag{2.24}$$

where high similarity indicates shared evolutionary dynamics, whereas lower similarity suggests a transition to a new variant or lineage. Thresholding or clustering on the similarity values $\{s_i\}$ thus provides a data-driven criterion for detecting community boundaries in the tree, without requiring manually defined thresholds or expert annotations.

In summary, contrastive learning provides a principled framework for learning phylogenetically meaningful embeddings. By combining self-supervised representation learning with downstream classification or clustering objectives, this approach enables automatic identification of lineages or variants characterized by distinct evolutionary or epidemiological behaviors, laying the foundation for scalable and adaptive genomic surveillance.

2.4 Analysis of the state-of-the-art

This section focuses on existing approaches to tackle the problem of lineage detection in phylogenetic trees.

2.4.1 Variant Hunters

The term "Variant Hunters" refers to molecular biologists or epidemiologists responsible for detecting and monitoring emerging pathogen variants through genomic analysis. Such genomic surveillance approaches have been crucial throughout recent outbreaks, where phylogenetic and phylodynamic tools enabled real-time tracking of viral diversification and the identification of emerging lineages [8, 21]. Following sample collection and sequencing, the pathogen's genome, which encodes all the information required for replication, is compared to previously sequenced genomes of the same species. When significant differences are observed, a new sequence may be classified as a mutation or, if associated with multiple distinctive changes, as part of a new variant. This process is exemplified in early genomic epidemiology studies, such as the surveillance of SARS-CoV-2 in Lombardy, where multiple co-circulating lineages were detected and monitored through systematic sequencing [3].

However, not all sequence differences correspond to biologically meaningful mutations. Each sample is transformed into a digital representation of the viral genome, typically consisting of approximately 30,000 nucleotides in the case of SARS-CoV-2. These genomic sequences are then aligned and compared to a reference genome, such as the original Wuhan strain, to identify substitutions, deletions, or insertions that have occurred during viral replication.

2.4.2 PhyloWave and other approaches

PhyloWave [9] is a computational framework designed to detect emerging pathogen lineages that display distinct growth dynamics within phylogenetic trees. The method aims to identify clades with increased fitness, which may indicate changes in transmissibility or immune escape. By analyzing local variations in lineage expansion over time, PhyloWave provides an early warning system for identifying variants of concern directly from genomic data.

Phylowave relies on an index computation for each internal or terminal node, based on a pathogen-specific kernel timescale, which weights more short distances to detect clades with increased fitness.

In empirical applications, PhyloWave has been successfully applied to four large-scale viral datasets: SARS-CoV-2, Pertussis, Tuberculosis and H3N2 (a subtype of influenza). In retrospective analyses, the method detected the rise of SARS-CoV-2 variants such as Alpha, Delta, and Omicron several weeks before their epidemiological dominance became apparent. It detected two additional undetected clades for Pertussis. This demonstrates the potential of tree-based dynamic analysis as an independent and quantitative surveillance tool to track the evolutionary fitness of circulating lineages. Phylowave pipeline is shown in fig. [9].

A recent line of work introduces statistical approaches for detecting variation in lineage fitness directly from phylogenetic trees. Volz and Didelot (2025) propose a model in which each branch is associated with a *coalescent propensity*, a heritable continuous trait quantifying the expected contribution of a lineage to future descendants. Variation in this trait reflects underlying selective differences, allowing the model to predict which lineages are expected to expand [24]. Their framework also provides a principled method to group branches into clusters with similar growth behavior, addressing a problem closely related to phylogenetic community detection [24].

2.4.3 Limitations of current paradigms

Although PhyloWave has introduced important progress in the detection of lineage-specific fitness shifts, current approaches remain constrained by several conceptual and practical limitations that hinder their scalability and generalization across pathogens.

The most fundamental limitation concerns their dependence on substantial domain knowledge and ad hoc parameterization. PhyloWave relies on expert-defined thresholds to determine when a clade's diversification rate deviates significantly from the expected neutral pattern. These thresholds, including the choice of time windows for local growth estimation, smoothing parameters for lineage-through-time curves, and statistical cutoffs for anomaly detection, must often be fine-tuned manually for each dataset. Small changes in these parameters can lead to markedly

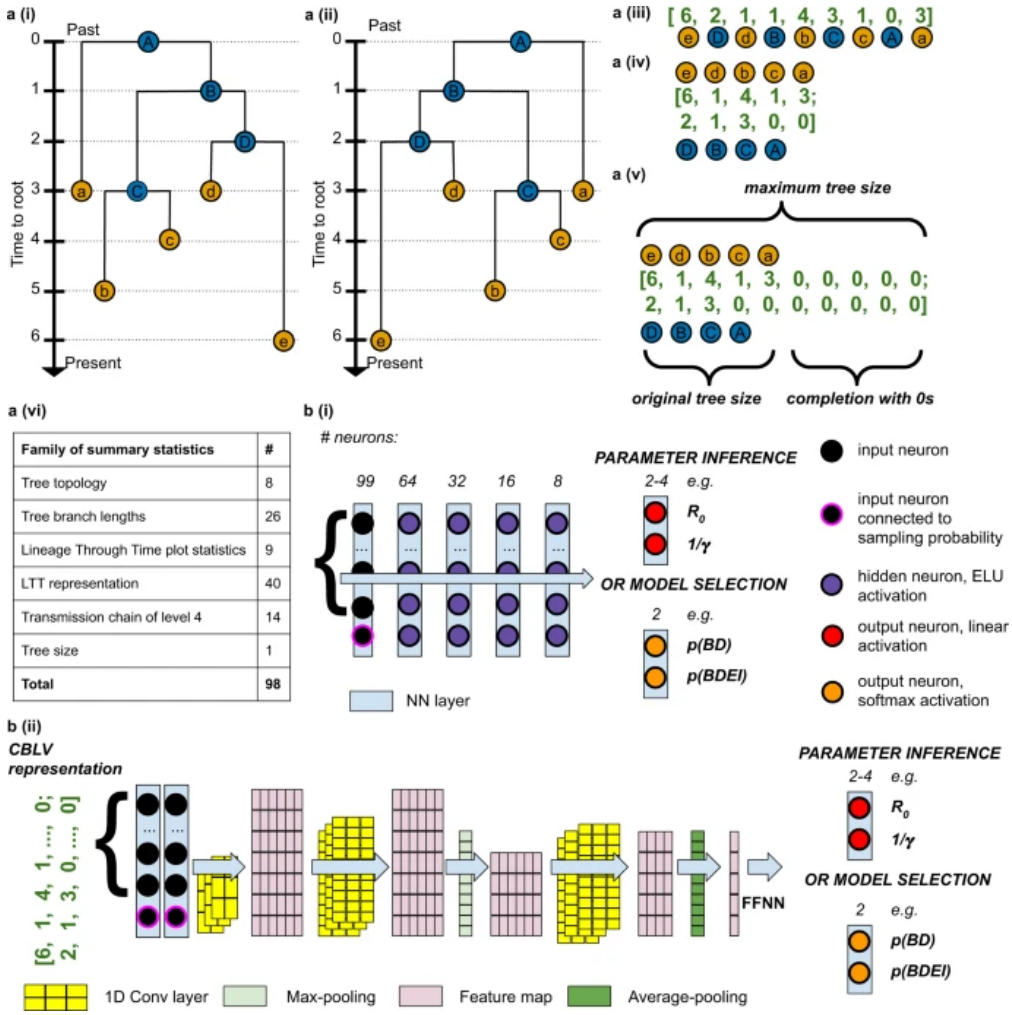
different outcomes, making the method sensitive to subjective decisions and difficult to reproduce. As a consequence, the approach cannot be applied automatically to new pathogens or outbreak scenarios without expert supervision.

Moreover, PhyloWave assumes the existence of a neutral birth–death baseline against which anomalous diversification is evaluated. In real-world epidemiological systems, however, transmission rates, sampling intensity, and selection pressures vary dynamically across time and space. When these assumptions are violated, apparent growth anomalies may arise purely from uneven sampling or transient demographic fluctuations rather than genuine fitness changes. This limits the interpretability and reliability of detected signals, especially in rapidly evolving viral populations [21].

Finally, the paradigm underlying PhyloWave is inherently heuristic: it depends on pre-defined statistical features of the tree and fixed decision criteria rather than learned representations. This rigidity prevents the model from adapting to novel or complex evolutionary scenarios that differ from those anticipated during design. Deep learning–based approaches such as PhyloDeep [10] and simulation-based neural estimators [22] demonstrate the advantages of flexible, data-driven inference strategies.

The line of work explored in [24] illustrates both the strengths and the limitations of statistical, non–machine-learning approaches: while theoretically grounded and interpretable, they require strong parametric assumptions and can become computationally demanding for large trees. These constraints motivate the need for scalable, data-driven models, such as recursive neural networks, that can learn clade structure directly from tree topology without relying on extensive manual calibration.

These limitations collectively motivate the need for a data-driven alternative that eliminates the dependence on manual thresholding and domain-specific heuristics. The framework proposed in this thesis addresses these issues by integrating contrastive representation learning with phylogenetic modeling. Instead of manually defining what constitutes a fitness change, the model learns from simulated multi-type birth–death processes in which mutation events alter lineage-specific transmission rates. Recursive neural networks encode the hierarchical structure of phylogenetic trees, while a contrastive objective ensures that lineages with distinct evolutionary dynamics occupy separate regions in latent space. This approach replaces ad hoc calibration with a principled learning paradigm, enabling scalable, automated, and pathogen-agnostic detection of fitness changes across evolving lineages.



Family of summary statistics	#
Tree topology	8
Tree branch lengths	26
Lineage Through Time plot statistics	9
LTT representation	40
Transmission chain of level 4	14
Tree size	1
Total	**98**

Figure 2.9: PhyloDeep pipeline. Tree representations: a (i), simulated binary trees. The simulations were encoded into two representations, either a (ii–v) with CBLV or a (vi) with summary statistics (SS). CBLV is obtained through a (ii) ladderization and a (iii) an inorder tree traversal. a (iv), an input matrix in which the information on internal nodes and leaves is separated into two rows. a (v), this matrix is padded with zeros so that the matrices for all simulations have the size of largest simulation matrices. SS consists of a (vi), a set of 98 statistics: 83 published in Saulnier et al., 14 on transmission chains and 1 on tree size. The information on sampling probability is added to both representations. b Neural networks are trained on these representations to estimate parameter values or to select the underlying model. For SS, b (i), a deep feed-forward neural network (FFNN) of funnel shape . For the CBLV representation, b (ii), convolutional neural networks (CNN). The CNN is added on top of the FFNN. Figure taken from [10]

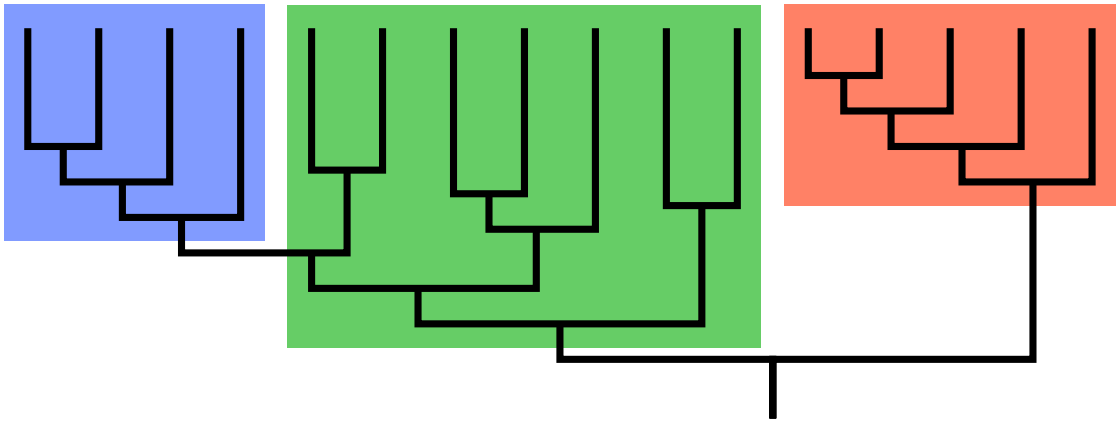


Figure 2.10: Example of a phylogenetic tree partitioned into three monophyletic clades (blue, green, red). Each colored region highlights a group of taxa sharing a recent common ancestor, illustrating how clades naturally define positive pairs (within the same color) and negative pairs (across different colors) for contrastive learning. Image adapted from the Wikipedia page "Clade" (`https://en.wikipedia.org/wiki/Clade`), licensed under CC BY-SA 4.0.

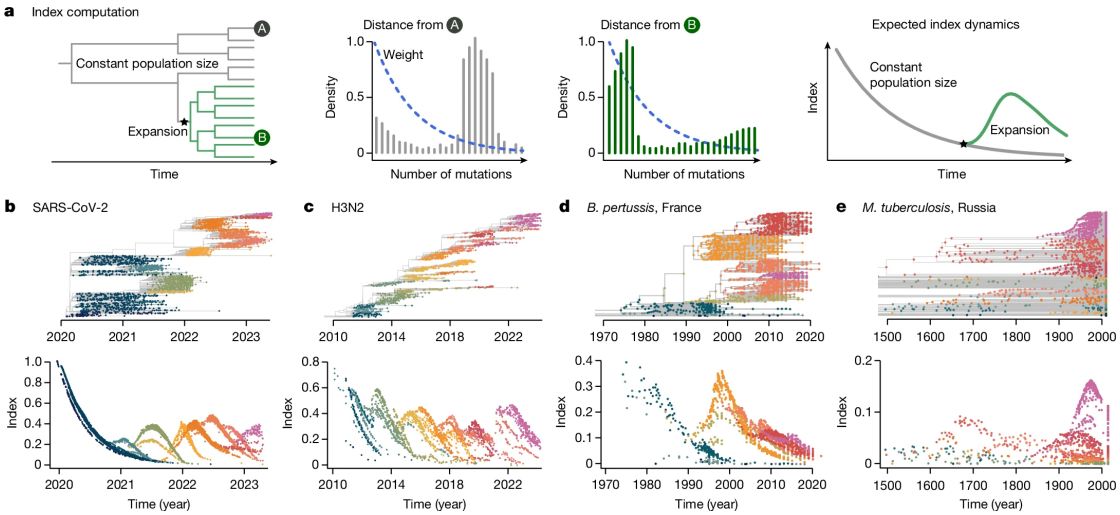


Figure 2.11: Phylowave pipeline. a, Schematics describing the principles of index computation. From left to right: example of a time-resolved phylogenetic tree with a background population (grey) and an emerging lineage (green); pairwise distance distribution from terminal node A, or terminal node B, respectively, to the rest of the population, with the dashed blue line denoting the geometric weighting; and expected index dynamics over time (see Methods for details). b–e, For each pathogen, SARS-CoV-2 (b), influenza A/H3N2 (H3N2; c), B.pertussis (d) and M.tuberculosis (e) the index dynamics computed at each node (terminal or internal) is presented. Colors represent the different lineages identified by their different index dynamics. Figure taken from [9]

Chapter 3

Materials & Methods

The subsequent sections are dedicated to the two main methodological contributions of this work: (i) adapted recursive neural networks and (ii) smoothed contrastive loss. Then, the materials are discussed. Following sections include materials, analysis and NN training and implementation details.

3.1 Contribution I - Adapted recursive neural networks

The Phyloscope encoder applies the concept of recursive neural networks (RvNNs) to the domain of phylogenetic trees. Recursive neural architectures are specifically designed to process hierarchical data structures, in which each internal node represents a composition of its children. Unlike conventional feed-forward networks that operate on fixed-length inputs, recursive networks propagate information upward through the topology of a tree, computing a vector representation for each node based on the representations of its descendants.

Formally, for a given node v with left and right children v_L and v_R, the encoding $\mathbf{h}_v$ is computed as a non-linear transformation of the encodings of its children, together with additional structural information such as branch length (Fig. 3.1) The Phyloscope implementation follows this general recursive formulation:

$$\mathbf{h}_v = f_\theta([\ell_v, \mathbf{h}_{v_L}, \mathbf{h}_{v_R}]), \tag{3.1}$$

where $[\cdot]$ denotes concatenation, ℓ_v is the branch length associated with node v, and f_θ is a learnable function parameterized by the neural network weights θ.

In practice, this function is implemented as a multi-layer perceptron (MLP) that receives as input the aggregated encodings of the child nodes together with the corresponding branch lengths. The MLP receives as input the concatenation of the children's encodings and the corresponding branch lengths. Its output is then

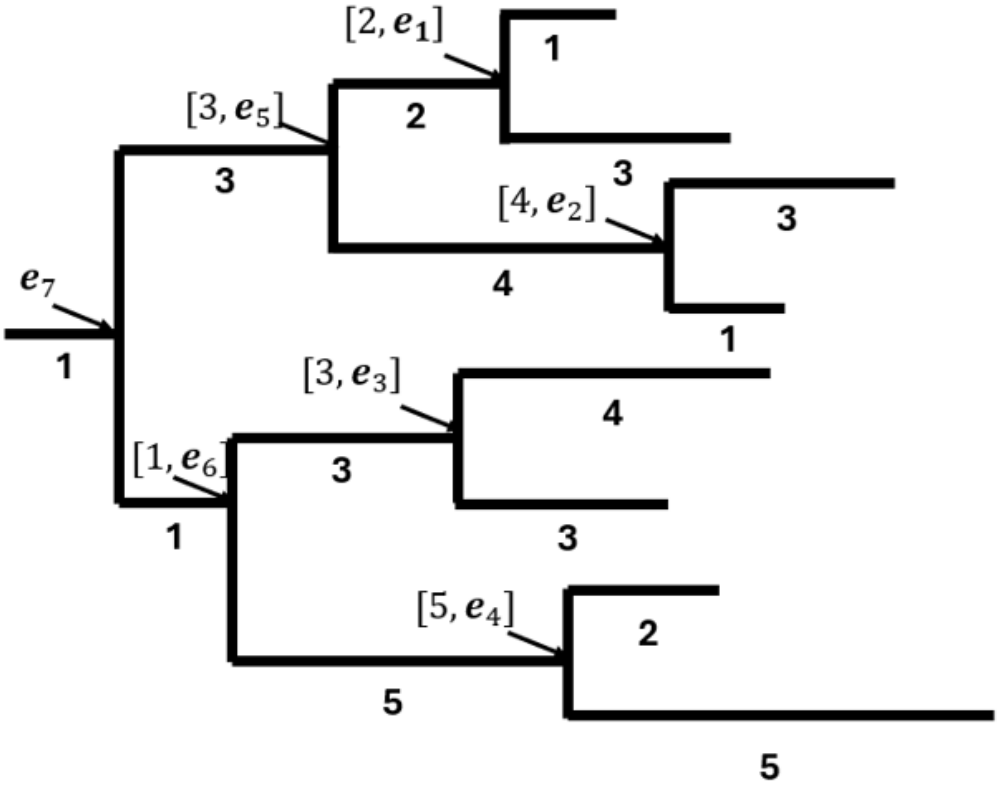


Figure 3.1: Example of a phylogenetic tree in which each internal node is assigned a vector representation computed by the MLP together with its branch length, except for the root, where only the final encoded vector is shown.

combined through an element-wise mean across the two child branches, producing the updated representation for the internal node.

The recursive process proceeds level by level, following the hierarchical organization of the tree. In the implementation, which is the core contribution of this thesis, this is achieved by iterating through the height levels of all nodes within a batch, starting from the lowest internal nodes and progressing toward the root. For each level, nodes whose children have already been encoded are selected, and their encodings are updated according to:

$$\mathbf{h}_v = \mathrm{MLP}\Big(\mathrm{concat}\Big(\ell_v, \bar{\mathbf{h}}_{\mathrm{children}}, \Big)\Big)\,, \tag{3.2}$$

where $\bar{\mathbf{h}}_{\mathrm{children}}$ is the mean of the children's encodings. This operation can be interpreted as a bottom-up message-passing mechanism that aggregates descendant information into a compact latent representation.

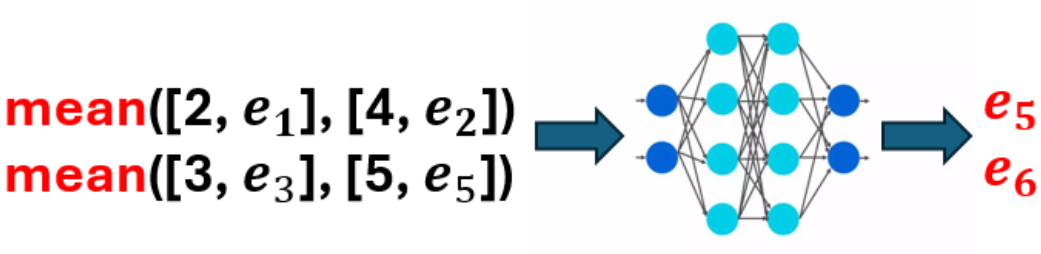


Figure 3.2: Illustration of the encoding step for internal nodes: for each node, the vectors corresponding to its left and right children (with respective branch lengths) are averaged, and the resulting mean vector is passed through the MLP to produce the internal representations e_5 and e_6.

All node encodings are updated in place within the batch tensors, ensuring computational efficiency and allowing the model to process multiple trees simultaneously. Once the recursion has reached the topmost level, each tree is represented by a hierarchy of learned encodings, culminating in a final root embedding that summarizes the entire phylogeny.

This adapted recursive neural network therefore provides a natural and theoretically grounded approach for learning from tree-structured data. By explicitly respecting the hierarchical dependencies among nodes, it enables the model to capture both local evolutionary events (encoded at lower levels) and global tree-wide patterns (emerging near the root). The resulting representations can then be exploited for downstream analysis, clustering, or evolutionary inference within the Phyloscope framework.

Input tree structure. The model assumes that each input phylogeny satisfies a set of structural constraints:

- **Binary topology:** every internal node must have at most two children. Trees that contain nodes with more than two descendants are considered invalid and are rejected during preprocessing. This constraint simplifies the representation of the tree as a matrix where each row corresponds to a node and each node stores at most two child indices.

- **Defined branch lengths:** all nodes must be associated with a branch length connecting them to their parent. Missing branch lengths are not allowed, as they form an integral part of the numerical representation. These values are later rescaled to achieve a comparable magnitude across different trees.

- **Unique node identifiers:** each node is required to have a unique name within a tree. This property allows unambiguous mapping between node-level labels, metadata, and learned embeddings.

Before any encoding takes place, each tree is systematically checked for compliance with these assumptions. Trees that fail these structural requirements are discarded, as they would violate the model's internal representation and could lead to inconsistent or misleading embeddings.

Branch length normalization. Branch lengths are used directly by the encoder, so it is important that they stay on a comparable scale across different trees. Since raw branch lengths can vary a lot (because of different reconstruction procedures or evolutionary models), each tree is rescaled so that its average branch length becomes 1.

For a tree with branch lengths $\{\ell_1, \ell_2, \ldots, \ell_K\}$, the mean value is

$$\bar{\ell} = \frac{1}{K} \sum_{i=1}^{K} \ell_i, \tag{3.3}$$

and every branch length is replaced by

$$\ell_i \leftarrow \frac{\ell_i}{\bar{\ell}}. \tag{3.4}$$

This simple normalization keeps the numerical scale stable and makes different trees easier for the model to compare.

Cluster definition and minimum leaves per cluster. To detect meaningful clusters in the tree, the embeddings computed at each internal node need to be related to a measure of similarity on the branches. For this reason, each internal node v has an embedding $\mathbf{h}_v$, and its similarity with the parent node $p(v)$ is computed.

Branch-level similarity. For every internal node v, the similarity between v and its parent is defined using cosine similarity:

$$\mathrm{sim}(v, p(v)) = \frac{\mathbf{h}_v^\top \mathbf{h}_{p(v)}}{\|\mathbf{h}_v\| \, \|\mathbf{h}_{p(v)}\|}. \tag{3.5}$$

This score tells us how similar the two embeddings are: high similarity means that the subtree under v is close (in embedding space) to its parent, while low similarity means that the branch marks a clearer separation.

Minimum leaves per cluster. Not every internal node is considered a valid cluster. A hyperparameter called *minimum leaves per cluster* sets the minimum

number of descendant leaves required for a node to count as a cluster. If leaves(v) is the number of leaves under node v, then v is a valid cluster only if

$$\text{leaves}(v) \geq L_{\min}, \tag{3.6}$$

where $L_{\min}$ is the chosen threshold. Nodes that do not reach this number of leaves are considered too small to be meaningful, and they are effectively merged into the nearest ancestor that satisfies the requirement. In the experiments, this hyperparameter was set to 17: clusters whose nodes culminated in at least 17 descendant leaves were considered, otherwise the cluster information was inherited recursively until the base nodes.

Role in cluster identification. Putting these two elements together, the network focuses only on internal nodes that (i) have enough descendant leaves and (ii) show a clear embedding similarity or difference with their parent. This avoids creating clusters that are too small and helps ensure that cluster boundaries are placed where both the tree structure and the embeddings agree.

CBLV encoding for children of base nodes. Before the recursive encoder is applied, the tree is first scanned to identify which internal nodes are large enough to be treated as potential cluster roots. This uses the *minimum leaves per cluster* hyperparameter: an internal node is considered a valid base node if it has at least $L_{\min}$ descendant leaves as detailed in previous paragraphs (Fig. 3.3).

Once the base nodes are identified, only their immediate children are assigned an explicit structural encoding using a CBLV representation. Each CBLV vector is padded to a fixed size of $E + 1$, where E is the dimensionality of the embeddings produced by the neural network (Fig. 3.4).

Encoding dimension. The hyperparameter *encoding dimension* E specifies the size of the embedding vector produced by the MLP for every internal node. This is the dimensionality of the space in which all node embeddings live, and it determines how much information the model can capture in a single vector. A larger value of E gives the model more capacity to represent structural differences between subtrees, at the cost of increased computation.

In our experiments, the encoding dimension is set to $E = 128$, which provides enough capacity for the embeddings to represent the structural patterns of the tree without becoming a computational bottleneck.

Node selection and hierarchical representation. The full phylogeny is thus reduced to a set of base nodes and their associated subtrees. The procedure can be summarized as follows:

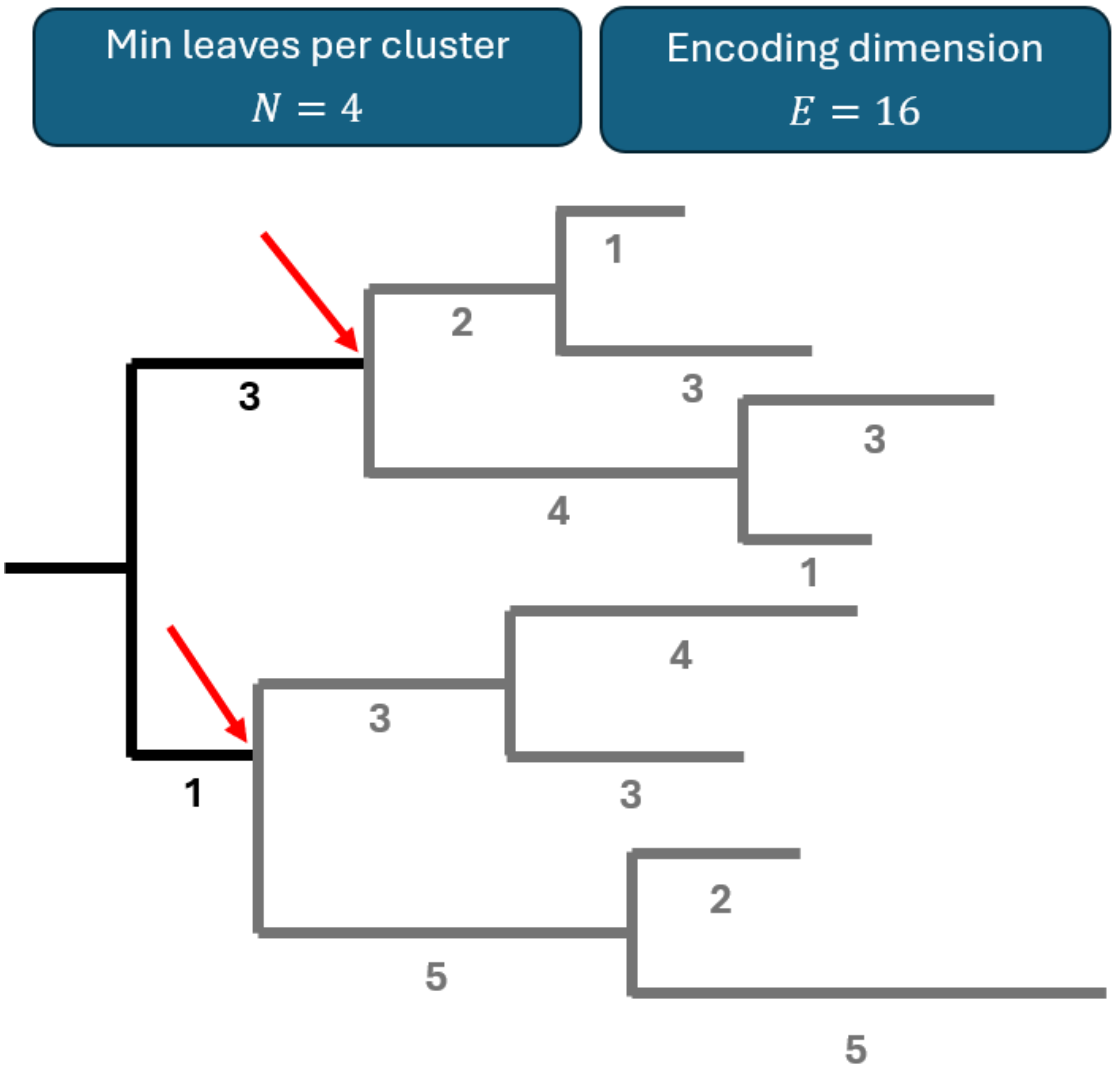


Figure 3.3: Example of a phylogenetic tree in which the internal nodes highlighted in red represent the base nodes selected according to the minimum–leaves criterion. For illustration purposes, the figure uses small hyperparameter values (e.g. minimum leaves per cluster and encoding dimension), which are not the ones used in the actual implementation.

1. Identify all internal nodes that satisfy the minimum leaf requirement (leaves(v) $\geq$ $L_{\min}$). These nodes define the roots of the smallest valid clusters.

2. For each base node, generate a CBLV encoding for its two children, padded to dimension $E + 1$.

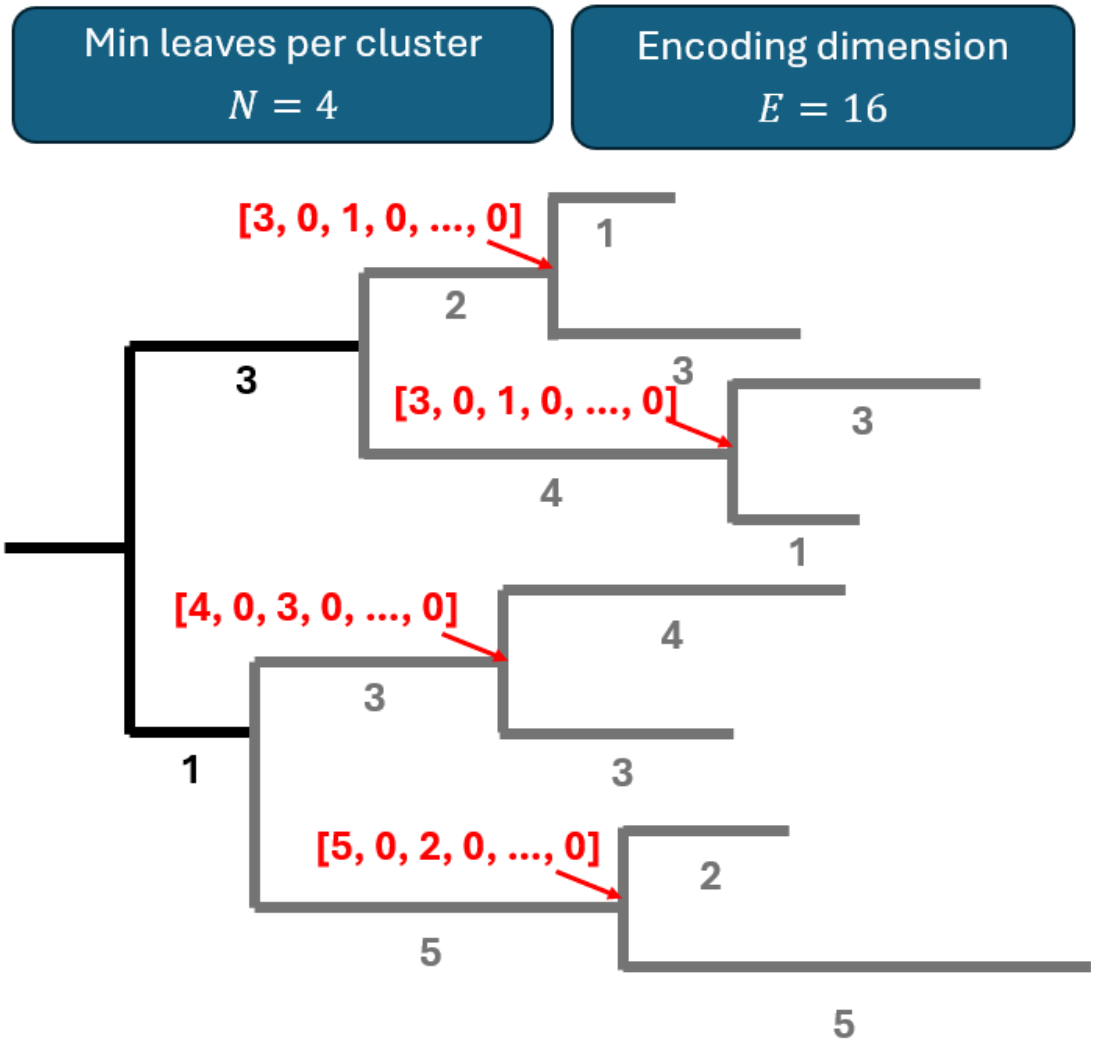


Figure 3.4: Example of the CBLV representations assigned to the nodes directly below the base nodes identified in the previous step. Each red vector shows the CBLV encoding padded to dimension $E+1$, assigned only to the immediate children of the base nodes, while the rest of the tree is processed recursively by the neural network.

3. All deeper nodes receive embeddings computed by the MLP during the bottom-up recursive pass.

This produces a clean hierarchical structure where explicit CBLV information is injected at the boundaries of each valid cluster, and the recursive neural encoder handles the rest of the tree structure.

3.2 Contribution II - Contrastive Loss for Community Detection

From recursive encodings to a learning objective. Once these embeddings are computed, the model needs a training objective that can make use of them and change nodes to organize in a meaningful way.

3.2.1 Smoothed Contrastive Loss

The second main contribution of this work is the definition of a contrastive loss function that uses these node embeddings to detect evolutionary communities inside the tree. This loss compares pairs of internal nodes within the same phylogeny and pushes the embeddings to reflect how close or far their evolutionary behavior is, according to the anchors associated with the nodes.

Given N encodings $\mathbf{x}_i \in \mathbb{R}^d$ and scalar anchors[1] $a_i \in \mathbb{R}$, the loss first L2-normalizes each encoding,

$$\hat{\mathbf{x}}_i = \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \tag{3.7}$$

then forms cosine similarities and the corresponding cosine distances

$$\mathrm{sim}_{ij} = \hat{\mathbf{x}}_i^\top \hat{\mathbf{x}}_j, \qquad d_{ij} = 1 - \mathrm{sim}_{ij} \in [0,2] \tag{3.8}$$

and considers only unordered index pairs (i, j) with $i < j$.

Anchor-conditioned target similarity. For each pair we define a soft target similarity

$$t_{ij} = \exp\Big(- \, s \, |a_i - a_j| \Big) \in (0,1] \tag{3.9}$$

where $s > 0$ is the *scale*. When $|a_i - a_j| = 0$ we have $t_{ij} = 1$, and as $|a_i - a_j|$ grows, t_{ij} decays toward 0 at a rate controlled by s.

Per-pair loss. In this setting, the indices i and j refer to internal nodes belonging to the same phylogenetic tree. Thus, each pair (i, j) compares two node encodings within a single tree, and the loss encourages nodes with similar anchor values to remain close in the embedding space while pushing apart nodes whose anchors differ substantially. The loss for a pair (i, j) is the mixture

$$\ell_{ij} = \underbrace{t_{ij}\, d_{ij}^2}_{\text{pull term}} + \underbrace{(1 - t_{ij}) \Big(\max\{0,\ m - d_{ij}\} \Big)^2}_{\text{push term}} \tag{3.10}$$

[1] the anchors are the logged birth rates of each mutation saved in the metadata and passed to the loss as additional information

with margin $m > 0$.

Total objective. Averaging over all $\frac{N(N-1)}{2}$ unordered pairs yields

$$\mathcal{L} = \frac{2}{N(N-1)} \sum_{1 \le i < j \le N} \left[t_{ij}\, d_{ij}^2 + (1 - t_{ij}) \left(\max\{0,\ m - d_{ij}\} \right)^2 \right] \tag{3.11}$$

Pull vs. Push. The *pull* term $t_{ij}\, d_{ij}^2$ dominates when anchors are close ($t_{ij} \approx 1$), penalizing the squared cosine distance and therefore encouraging high cosine similarity between $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_j$. The *push* term $(1-t_{ij}) \left(\max\{0,\ m-d_{ij}\} \right)^2$ dominates when anchors are far ($t_{ij} \approx 0$), acting as a squared hinge that is active only for $d_{ij} < m$ and driving pairs apart until the margin m is reached; for $d_{ij} \ge m$ the hinge is inactive and contributes no penalty.

Role of the scale s and of t_{ij}. The scale s controls how sharply t_{ij} transitions from 1 to 0 as $|a_i - a_j|$ increases: small s yields a gradual blend of pull and push over a wider anchor gap, while large s yields a rapid switch, making the loss more sensitive to anchor differences. Thus t_{ij} serves as a data-driven weight that interpolates continuously between attraction and repulsion, aligning the geometry of the normalized embeddings with the geometry induced by the anchors.

Ranges and practical choice of m. Because vectors are L2-normalized, $\mathrm{sim}_{ij} \in [-1,1]$ and $d_{ij} \in [0,2]$, which makes $m \in [0,2]$ a natural range; for dissimilar pairs ($t_{ij} \approx 0$) the margin implies a target maximum cosine similarity of $1 - m$.

3.3 Materials

This section is dedicated to the materials used in this work.

3.3.1 MTBD-based mutation model

In the Multi-Type Birth-Death (MTBD) model framework used in this thesis, lineages undergo four types of events over time: **birth**, **death**, **mutation**, and **sampling**.

SARS-COV-2 variants, among many other pathogens, have gained wide-spread transmission due to their adaptability and infectiousness, which resulted in increased fitness and virus transmission at a rapid rate [3]. To account for this process, the

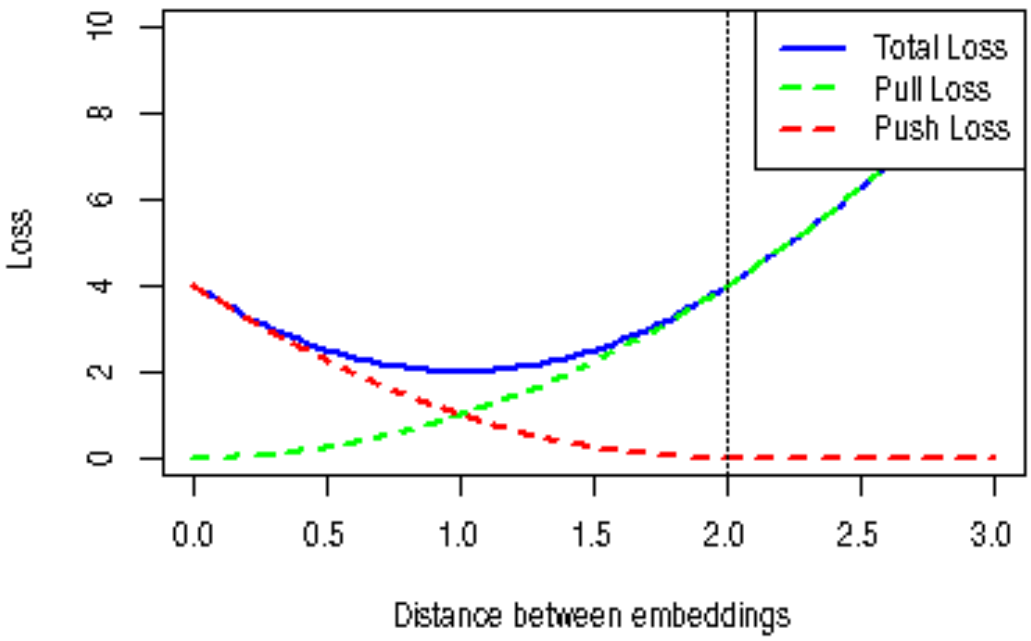


Figure 3.5: Plot of the smoothed contrastive loss showing its three components: the pull loss (green), which penalizes small distances for similar-anchor pairs; the push loss (red), which penalizes distances below the margin m for dissimilar-anchor pairs; and the total loss (blue), given by their weighted combination as defined in Eq. (3.10). The vertical dotted line marks the margin m.

library `phylogenie`[2] extends the MTBD framework with a separate **mutation process**, which records new variants of existing states over time.

Mutation events in `phylogenie` In `phylogenie`, mutations are modeled as independent stochastic events. Mutation events are *migration* events, which change the epidemiological state of a lineage. For example, suppose some nodes of the phylogenetic tree have undergone two mutations:

[2] `phylogenie` is a Python library developed to simulate a large number of trees; this tool turned out to be incredibly useful for the experiments presented throughout this thesis.

`I` → `MUT-1.I` → `MUT-2.I`

`MUT-1.I` and `MUT-2.I` represent two mutations with increased birth rate with respect to `I`(Fig. 3.7).

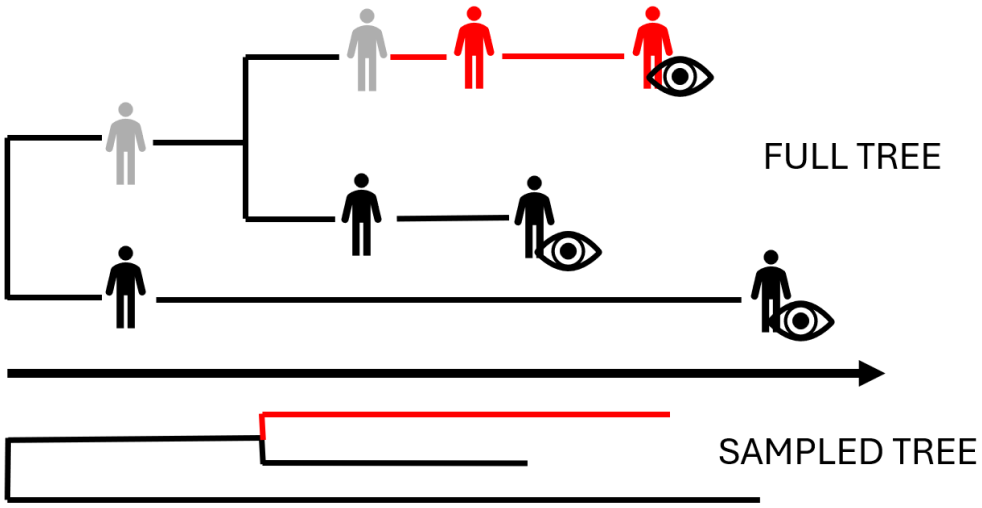


Figure 3.6: Illustration of the full evolutionary tree (top) and the corresponding sampled tree (bottom).Red segments mark the occurrence and propagation of a mutation, while eye symbols indicate sampling events. Only the sampled lineages and the minimal structure connecting them are retained in the reconstructed tree.

The extended MTBD mutation framework implemented in `phylogenie` relies on the Gillespie Stochastic Simulation Algorithm (SSA) to generate exact realizations of the underlying continuous-time Markov process. In this setting, lineages evolve by undergoing discrete events in continuous time, such as *birth*, *death*, *sampling*, or *mutation*, each characterized by an instantaneous rate.

As illustrated in Fig. 3.6, the sampled phylogeny corresponds to the type of tree representation produced by this simulation framework.

At any given time, let

$$E = \{e_1, e_2, \ldots, e_{N_e}\} \tag{3.12}$$

be the set of all possible events. Each event e_i occurs at a rate r_i and can act on a population of size S_i. The *propensity* of event e_i is therefore defined as:

$$p_i = r_i \times S_i \tag{3.13}$$

The total rate at which any event occurs is the following:

$$\lambda = \sum_{i=1}^{N_e} p_i \tag{3.14}$$

Gillespie's algorithm proceeds in two steps:

1. **Sampling the waiting time.** The time to the next event is drawn from an exponential distribution with rate λ:

$$\Delta t \sim \text{Exp}(\lambda),$$

ensuring that events occur in continuous time with statistically correct spacing.

2. **Selecting the event.** Once the waiting time is sampled, the next event is chosen using weighted sampling, where the probability of selecting event e_i is:

$$\Pr(e_i) = \frac{p_i}{\lambda}.$$

If the event acts on individual lineages (e.g., mutation or birth), one lineage is then selected uniformly at random among the S_i eligible ones.

This procedure is repeated until the sampled tree reaches the desired number of tips.

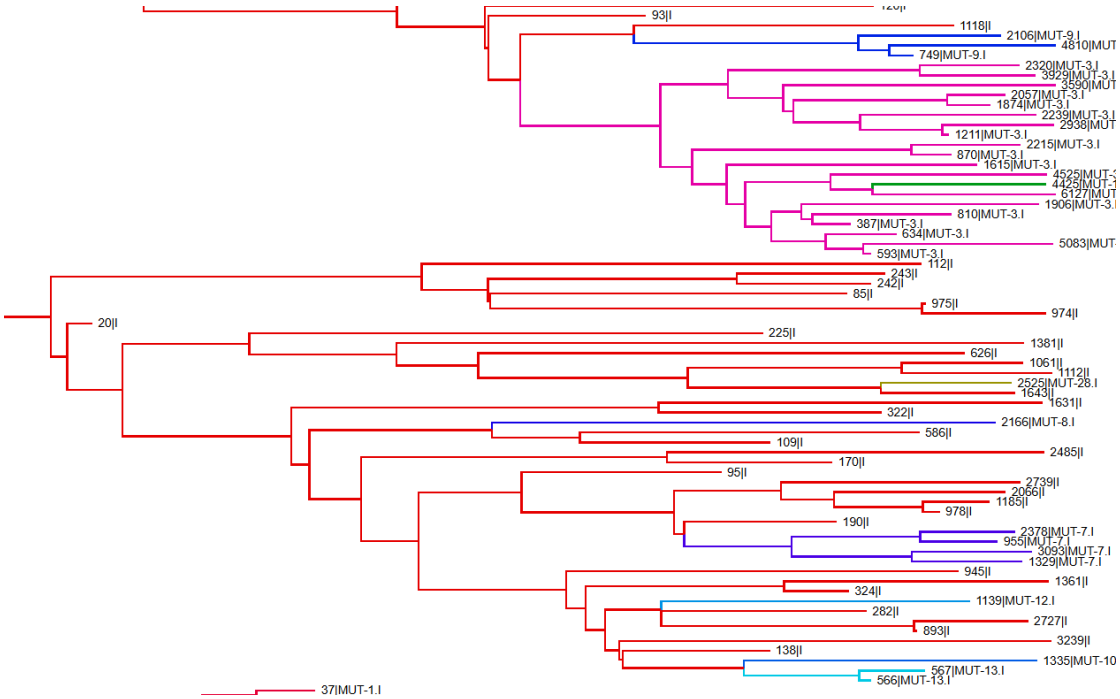


Figure 3.7: Portion of a simulated tree. In red, nodes with no mutation are represented. Each color identifies a new mutation, where the birth rate is increased.

In Fig.3.7[3], several mutations are identified by different colors. When a mutation occurs, a mutated individual inherits the same events as the original lineage, except for the birth rate. The birth rate of this mutated variant is obtained by multiplying the previous rate by a random variable $S^{(k)}_{\text{birth}}$, called the *birth-rate scaler*:

$$\lambda_{\text{MUT}_k.i} = S^{(k)}_{\text{birth}} \times \lambda_i \tag{3.15}$$

The variable $S^{(k)}_{\text{birth}}$ is drawn from a probability distribution $\mathcal{D}_{\text{birth}}$.

3.3.2 Simulation data

To evaluate the proposed framework under controlled yet realistic evolutionary scenarios, a large collection of synthetic phylogenetic trees was generated using a stochastic birth-death process with mutations. This simulation setup allows direct control over key epidemiological and evolutionary parameters, while preserving the complex branching structure characteristic of real-world phylogenies.

A total of **100,000 independent trees** were simulated. Each tree was generated under a birth-death parameterization, in which the reproduction number, infectious period, sampling proportion, and mutation rate are either fixed or drawn from predefined probability distributions.

Birth-death parameterization. The underlying model follows a standard birth-death formulation, as detailed in 2. In this setting, the process is parameterized through the *basic reproduction number* R_0, which specifies the expected number of secondary infections produced by a single infectious lineage in the birth–death model. In this setting, the infectious period was fixed at **2.5 time units**, which simply sets the rate at which infectious lineages stop producing new infections in the simulation.

Distributions of simulation parameters. Table 3.1 summarizes the probabilistic specification used to generate the simulation parameters. All parameters are sampled independently for each tree from the following distributions:

- The reproduction number R_0 is drawn from a log-normal distribution with mean 1 and standard deviation 0.2 on the log scale.
- The sampling proportion s is sampled from a uniform distribution on the interval $[0.1, 1.0]$.

[3]the picture was taken from `Icytree`, a web-based tool to visualize trees, color them or gather statistics

- The mutation rate r is drawn from a uniform distribution on $[0.01, 0.1]$.
- The birth-rate scaling factor used during mutation events is drawn independently *for each mutation event* from a normal distribution with mean 2 and standard deviation 0.2.

Table 3.1: Probabilistic specification of the main simulation parameters.

Parameter	**Distribution**	**Hyperparameters**
Effective reproduction number R_0	Log-normal	mean $= 1,\ \sigma = 0.2$
Sampling proportion s	Uniform	low $= 0.1,$ high $= 1.0$
Mutation rate r	Uniform	low $= 0.01,$ high $= 0.1$
Birth-rate scaler (for mutation process)	Normal	$\mu = 2,\ \sigma = 0.2$

Sampling constraints and tree size. To ensure that all simulated trees contain sufficient information for meaningful analysis, explicit constraints were imposed on the number of observed tips. The process was conditioned to produce trees with between 200 and 500 tips (inclusive). The choice of this range follows PhyloDeep [10], where the BD simulations for large trees fell within the same interval.

Node-level features. In addition to the global parameters described above, the simulation records a set of features at the node level, which are later used during preprocessing and encoding. For every node in each tree, the following quantities are stored:

- a mutation-related label indicating the presence or identity of mutations,
- the absolute height of the node in the tree (distance from the root),
- a discretized height level, obtained by grouping nodes according to their height,
- the depth, defined as the distance from the root or the number of edges along the path from the root,
- the number of descendant leaves (subtree size), which is crucial for identifying nodes that qualify as clusters.

These node-level descriptors provide a rich structural and evolutionary context that can be exploited by the encoder. For example, the number of descendants is used to determine which nodes represent sufficiently large clusters, while depth

and height information contribute to the construction of the base encodings fed to the neural network. Furthermore, these terms contribute directly to data analysis through detailed statistics.

Reproducibility and implementation. The simulation of 100 000 trees was performed using a fully automated pipeline with a fixed random seed (`seed = 42`) to guarantee reproducibility of the generated dataset. The simulations were distributed across all available CPU cores (`n_jobs = -1`), and each individual simulation was subject to a time limit to avoid pathological runs that might fail to terminate within a reasonable time window. Unfortunately, the timeout affects the resulted simulated trees, as it depends on number of CPU cores used in the simulation.

3.3.3 Dataset

The simulated phylogenetic trees described in the previous subsection are transformed into a structured dataset that can be directly used for training the neural encoder. Each individual tree is converted into a self-contained sample that encapsulates both its topology and its associated evolutionary annotations. These samples are then stored to disk and later combined into mini-batches for efficient processing during training.

From simulated trees to structured samples. The tree is normalized by rescaling its branch lengths to have unit mean, ensuring numerical comparability with the rest of the dataset.

Each accepted tree is then converted into a structured representation consisting of:

- a list of node identifiers that uniquely label every node in the tree,
- a pair of indices for each node indicating its left and right child (or a special placeholder for leaves),
- a vector of branch lengths, one for each node,
- node-level attributes such as height, discrete height level, depth, and number of descendant leaves,
- cluster labels and cluster anchor values derived from the tree-level metadata.

This representation bridges the gap between the raw phylogenetic object and the tensor-based format required by the neural network.

Cluster labels and anchors. Cluster labels are used to assign each node to an evolutionary cluster. A node is considered a valid cluster only if the subtree beneath it contains at least $L_{\min} = 17$ descendant leaves; this is the same cluster-size threshold introduced earlier. Nodes that do not reach this threshold are merged into a larger clade so that clusters are not too small or unstable.

Once the clusters are defined, each of them is given an *anchor value* taken from the tree metadata (e.g., the birth rate associated with a mutation). These anchor values are later used by the contrastive loss to pull or push the node embeddings to reflect how similar or different the clusters are.

Base encodings and node features. For each node in the tree, a base encoding vector is constructed prior to any neural processing. This vector summarizes the local structure of the subtree rooted at that node, with a particular emphasis on depth-related information. A typical base encoding is obtained by considering the depths of the nodes encountered in a traversal of the subtree and expressing them relative to the depth of the current node. The resulting sequence is then padded with zeros up to the fixed encoding dimension used by the model.

In addition to these base encodings, the dataset retains the raw node features produced by the simulator, namely:

- indicators of mutation events or mutation-related labels,
- the continuous height of the node from the root,
- a discretized height level that groups nodes into height bands,
- the depth of the node, expressed as a path length from the root,
- the number of descendant leaves, which plays a central role in determining cluster membership.

These quantities are not all directly fed into the neural network, but they inform both preprocessing decisions (e.g., which nodes qualify as clusters) and the construction of the base encodings that serve as inputs to the encoder.

Construction of training batches. Although the fundamental unit of the dataset is the individual tree sample, training is performed on mini-batches of trees to leverage parallel computation. To construct a batch, several samples are selected and their node-level tensors are concatenated along the node dimension. A cumulative offset is applied to all child indices to ensure that references to children remain correct in the concatenated representation. In addition, a list of pairs is maintained to map each node in the batch back to its original tree and node identifier.

The resulting batched dataset thus consists of:

- a collection of normalized and structurally validated trees,
- for each tree, a complete set of node-level base encodings and structural descriptors,
- cluster labels and anchors suitable for contrastive learning,
- and a memory layout optimized for efficient processing on modern hardware.

3.3.4 Empirical data

The network was additionally evaluated on four real-world phylogenetic datasets corresponding to major viral and bacterial pathogens: *SARS-CoV-2*, *Mycobacterium tuberculosis* (tuberculosis), *Bordetella pertussis*, and influenza A subtype *H3N2*. These pathogens were not chosen arbitrarily. Their selection was strongly motivated by the recent Phylowave framework [9].

Real-world datasets By adopting the same set of pathogens, this thesis aligns its empirical evaluation with an emerging standard in the field. However, a direct comparison with Phylowave results is still not possible. Because Phyloscope has been trained on relatively small simulated trees, it cannot handle the full pathogen trees. For this reason, preprocessing and subsampling become essential steps in the pipeline, ensuring that the real-world phylogenies satisfy both simulated trees requirements and are small enough to test the validity of this network.

Preprocessing & Subsampling Before being used for model evaluation, each empirical phylogeny underwent a standardized preprocessing procedure to ensure structural consistency and biological comparability with the simulated data. The overall objective of this step was to transform the heterogeneous raw phylogenies into curated, binary, and temporally constrained trees compatible with the Phyloscope framework.

Since phylogenetic reconstructions can contain polytomies (nodes with more than two descendants), an initial check verified whether the tree was strictly binary. In cases where polytomies were detected, they were recursively resolved by introducing minimal artificial branch lengths (`1e-6`) to preserve topological integrity. This ensured that every internal node had exactly two children, which is a strict requirement for the neural encoder, without impacting on Phyloscope performances.

Once binary, the tree was *ladderized* according to the number of descendant leaves (`n_leaves`), resulting in a consistent left-right ordering of nodes. Branch lengths were rescaled to normalize the tree and remove scale-dependent biases among datasets.

Each phylogeny was associated with a corresponding metadata file containing leaf-level attributes such as isolate identifiers, sampling dates, and clade designations. These metadata files were parsed and cross-referenced with the leaf identifiers in the tree to attach the appropriate information to each node. Depending on the dataset, different metadata columns were used to extract clade and time information, including `Genotype`, `Clade`, `Nextstrain_clade`, `Global_clade` for cluster-specific information and `Collection_time` for time information, useful for the subsampling step.

A dataset-specific filtering step was then performed to obtain subsampled trees that were comparable in size to the simulated training trees. The ideal target range was between 200 and 500 tips, matching the interval used in the simulated dataset. Values slightly below 200 were also accepted when necessary, since some pathogens did not contain enough samples to meet the lower bound while still preserving meaningful clade diversity, and alternative subsampling strategies produced too many tips. Trees close to (but below) 500 tips were considered acceptable, as they remained manageable for the Phyloscope architecture.

Table 3.2: Overview of the empirical datasets and corresponding subsampling criteria. The selection aimed to produce trees with a number of tips comparable to the simulated range (less than 500 tips), where possible the >200 tips was respected

Pathogen	**Selection criterion**	**Time window**
TB	"Euro" clades	—
SARS-CoV-2	Samples collected in 2023	2023
Pertussis	Samples from 2009–2011	2009–2011
H3N2	Samples collected in 2022	2022

To ensure consistent labeling across all levels, clade information was propagated upward from the leaves to their parent nodes in post-order traversal, allowing internal nodes to inherit the clade identity of their left descendant. This propagation enabled full clade annotation across the tree.

Through this standardized preprocessing and subsampling pipeline, all empirical phylogenies were made structurally and numerically comparable to the simulated data, thereby enabling a direct, unbiased evaluation of the Phyloscope neural framework on real-world evolutionary datasets.

3.4 Analysis & NN training

This section provides an overview of the analytical and computational steps required to validate the training data, construct the neural architecture, and optimize the Phyloscope encoder.

The section begins with an assessment of the simulated training data, verifying that the trees generated under the birth-death process possess the expected structural and evolutionary properties and are broadly comparable to empirical phylogenies. Subsequent subsections describe the data analysis procedures applied to both simulated and real-world trees, focusing on the extraction of relevant statistics and the preparation of neural-ready representations.

The architecture of the recursive encoder is then detailed, with particular attention to the design of the MLP used to process hierarchical information within each tree. Finally, the optimization strategy adopted during training is presented, including the choice of optimizer, learning rate, batch construction, and regularization techniques.

Together, these components define the methodology used to train Phyloscope and assess its ability to learn consistent, biologically meaningful representations of phylogenetic structures.

3.4.1 Validation of training data & Empirical data

To quantitatively assess the quality and internal consistency of the simulated dataset, a comprehensive set of structural and topological statistics was computed on a representative subset of the data. Specifically, the analysis was performed on the first 100 phylogenetic trees out of the 100,000 total simulations. This sampling strategy was adopted to obtain an interpretable yet statistically meaningful overview of the dataset characteristics, while maintaining computational efficiency given the large total number of trees.

The selected metrics were designed to evaluate both the *shape* and the *diversity* of the simulated trees, as well as the distribution of biologically relevant quantities such as mutation counts, cluster structure, and tree balance. For each of the 100 sampled trees, the following measures were calculated:

- **Tip depths:** the minimum and maximum depth of the leaves, providing a measure of the temporal or evolutionary span represented in each phylogeny.
- **Tree height and height levels:** indicators of the overall tree scale and the hierarchical structure across internal nodes.
- **Number of mutations and true labels:** quantifying the diversity of mutational states and the effective number of clusters retained after filtering by the minimum leaf threshold (`min_tips_per_cluster = 17`).
- **Difference between mutation and label counts:** highlighting discarded or merged clusters that did not meet the minimum size requirement.

- **Discarded label sizes:** the size distribution of clusters removed during preprocessing, giving insight into how many small subclades were pruned from the dataset.

- **Gini index of true labels:** a normalized measure of label imbalance across the leaves, ranging from 0 (completely homogeneous) to 1 (maximally diverse).

- **Sackin index:** a classical measure of tree balance, computed here in its normalized form to compare topological symmetry among trees.

- **Mean leaf pairwise distance:** the average distance between all pairs of leaves, reflecting the degree of diversification within each phylogeny.

- **Sampling proportion (`s`):** extracted from the simulation metadata, representing the fraction of the total population that was sampled in each simulation instance.

All statistics were computed using the `phylogenie` library and aggregated across the 100 sampled trees. The resulting distributions were visualized as histograms for each metric, enabling rapid inspection of potential outliers, biases, or structural anomalies. Although computed on a limited subset, the obtained results provide a reliable quantitative overview of the variability and stability of the simulated dataset, confirming that it adequately spans the parameter space intended by the birth–death model configuration.

To further evaluate the realism of the simulated trees and their suitability as training data, a comparative analysis was conducted between the simulated and empirical phylogenies. A set of common structural metrics was computed for both groups, allowing a direct, quantitative comparison of tree topology, size, and diversification patterns. Specifically, for each tree the following quantities were calculated:

- **Depth levels:** measures of overall tree scale and hierarchical structure;

- **Number of leaves:** reflecting the sampling density and effective tree size;

- **Sackin index:** quantifying tree balance and symmetry in branching patterns;

- **Mean leaf pairwise distance:** capturing the average evolutionary distance among sampled tips.

Specifically, results of this comparison will be discussed in 4.

3.4.2 Network Architecture

The neural component of Phyloscope is based on a feed-forward multi-layer perceptron (MLP) used as the core transformation module within the recursive encoding framework. At each internal node, the MLP receives as input the encodings of the two child nodes together with their branch lengths, and outputs a latent representation for the parent. The parent branch length is then appended afterward, producing the final encoding of the node.

Child representations. Each node v maintains two vectors: a latent encoding $\mathbf{e}_v \in \mathbb{R}^E$ and a full encoding $\mathbf{h}_v \in \mathbb{R}^{E+1}$ that includes the branch length ℓ_v. For each child v_L and v_R, a full representation is constructed as:

$$\mathbf{z}_{v_L} = \Big[\, \ell_{v_L},\ \mathbf{e}_{v_L} \,\Big], \qquad \mathbf{z}_{v_R} = \Big[\, \ell_{v_R},\ \mathbf{e}_{v_R} \,\Big], \tag{3.16}$$

where both vectors lie in $\mathbb{R}^{E+1}$.

These vectors are then aggregated to obtain a single representation summarizing the subtree below v:

$$\bar{\mathbf{z}}_v = \frac{1}{2}\Big(\mathbf{z}_{v_L} + \mathbf{z}_{v_R}\Big) \in \mathbb{R}^{E+1}. \tag{3.17}$$

MLP transformation. The aggregated vector $\bar{\mathbf{z}}_v$ serves as the input to the MLP, which computes the latent encoding of the parent node. The transformation is given by:

$$\mathbf{e}_v = \sigma(\mathbf{W}_2\, \sigma(\mathbf{W}_1 \bar{\mathbf{z}}_v + \mathbf{b}_1) + \mathbf{b}_2)\,, \tag{3.18}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are learnable weight matrices, $\mathbf{b}_1$ and $\mathbf{b}_2$ are the associated bias vectors, and σ denotes the ReLU activation function. The output $\mathbf{e}_v$ has dimension E.

Final parent encoding. The full encoding of v is obtained by concatenating its own branch length *before* the latent vector:

$$\mathbf{h}_v = \Big[\, \ell_v,\ \mathbf{e}_v \,\Big] \in \mathbb{R}^{E+1}. \tag{3.19}$$

Recursive propagation. Applying this procedure bottom-up across the tree allows the encoder to propagate structural, topological, and branch-length information from the leaves toward the root. The MLP acts as a shared, depth-invariant transformation at each internal node, enabling the model to learn consistent hierarchical representations that reflect both local patterns and global phylogenetic structure.

3.4.3 Optimization

The optimization of the Phyloscope encoder aimed to refine the parameters of the MLP and recursive framework in order to produce stable and biologically consistent node embeddings. Training was conducted using the **AdamW optimizer**, an improved variant of the classical Adam algorithm that decouples weight decay from gradient-based parameter updates. In traditional Adam, the L_2 regularization term is applied directly within the gradient update, which can lead to an incorrect interaction between the adaptive learning rate and the weight decay term. AdamW addresses this limitation by performing weight decay as a separate operation, ensuring a more accurate and theoretically sound form of regularization. As a result, the optimizer provides faster convergence, better generalization, and improved stability across deep and recursive architectures.

Training was performed in mini-batches of 64 samples, each batch containing multiple preprocessed phylogenetic trees. The dataset was randomly divided into a training set (80%) and a validation set (20%), ensuring a balanced representation of trees of different sizes and topologies across both subsets. A constant learning rate of 0.001 was adopted, which yielded smooth convergence without oscillations or divergence across epochs. A dropout rate of 0.5 was applied between hidden layers to reduce overfitting and promote robust generalization to unseen phylogenies.

The model was trained for a fixed number of epochs, with continuous monitoring of the validation loss to assess convergence and detect potential overfitting. The optimization process was terminated once the validation loss reached a plateau, indicating that the encoder had achieved a stable and generalizable set of parameters.

Through this combination of adaptive learning, decoupled regularization, and dropout-based noise injection, the training procedure achieved both numerical stability and robust convergence. This ensured that the final learned representations captured consistent topological patterns across simulated and empirical trees, validating the reliability of the training process.

Table 3.3: Summary of the main optimization hyperparameters used during training.

Parameter	Value / Description
Optimizer	AdamW (decoupled weight decay)
Learning rate	0.001 (constant across epochs)
Weight decay	Decoupled, default PyTorch implementation
Batch size	64 phylogenetic trees per batch
Training / Validation split	80% / 20%
Dropout rate	0.5 between hidden layers
Number of epochs	Fixed; training stopped at validation plateau
Loss monitoring	Validation loss tracked after each epoch
Early stopping criterion	Convergence of validation loss
Regularization strategy	Dropout + weight decay via AdamW

3.5 Implementation & Computation

This section is dedicated to implementation details and computational settings.

3.5.1 Model requirements

The computational framework developed in this work is designed to learn vector representations of phylogenetic trees through a neural network encoder that operates directly on tree-structured data. To guarantee both mathematical correctness and computational efficiency, a number of structural, numerical, and software requirements must be satisfied by the input data and by the execution environment.

Software and computational environment. All experiments were performed in `Python`, using `PyTorch` as the main deep learning library for tensor operations and automatic differentiation. Additional packages were used for specialized tasks: a phylogenetic toolkit to parse and manipulate trees in `Newick` format, a columnar data-processing library for metadata tables, and a parallelization library to accelerate sample preparation across multiple CPU cores.

Empirical data used in this thesis were obtained from the Supplementary Material of Phylowave paper [9], which obtained them, in turn, from Nextstrain, an open-source platform for real-time pathogen genomic surveillance. The phylogenies used for the empirical evaluation originate from the supplementary material of the *Phylowave* study, distributed as `.nexus` files containing both the inferred trees and associated metadata. These were converted to `Newick` format for compatibility with the encoder.

To ensure reproducibility and portability across computational systems, all experiments were executed inside containerized environments using **Singularity** and **Apptainer**. Long-running processes on the university servers were managed using `screen`, allowing training and simulation jobs to continue after detaching from the session.

Hardware resources. This work was conducted on two main computational infrastructures:

- **Politecnico di Torino departmental server**: 24 CPU cores, used primarily for initial simulations and preprocessing steps.
- **HPC cluster of Politecnico di Torino**: up to 96 CPU cores available for this thesis, used for large-scale simulations, sample generation, and training.

No GPU was required for training, as the model is lightweight and efficiently runs on CPUs.

Computation times. The full pipeline involves three main stages, simulation, sample generation, and model training, each executed only once. The corresponding computation times were the following:

- **Simulation of 100,000 phylogenies** (training + validation): approximately **2 days** on 96 CPU cores.
- **Creation of training and validation samples**: approximately **1.5 days**.
- **Training + validation** for 20 epochs on 100,000 trees: approximately **18 hours**.

These timings refer to the full-scale configuration adopted in this thesis and reflect the computational cost of the entire learning pipeline.

Chapter 4

Results

This chapter is dedicated to analysis of the results obtained in two settings: validation set and inference on empirical data, with an introduction to the results of the tree analyses, to the evaluation metrics used in this work and to the choice of threshold for cluster visualization.

4.1 Dataset Metrics Analysis

Before training and evaluating Phyloscope, both simulated and empirical phylogenies were characterized using the structural metrics described in Section 3.4.1. The aim of this analysis was to assess whether the simulated birth–death trees span a similar range of shapes, depths, and imbalance profiles as the empirical phylogenies. Ensuring this similarity is crucial, since the encoder should not be trained on unrealistic or overly homogeneous tree shapes.

Table 4.1 summarizes the main metrics computed for each empirical dataset. Although different pathogens exhibit distinct evolutionary behaviors, the collected values fall well within the variability observed across the 100,000 simulated trees used for training.

The following table shows different metrics gathered for empirical trees:

Table 4.1: Summary of phylogenetic metrics for each virus (simulated data excluded).

Pathogen	**Sackin**	**MLPD**	**Depth levels**	**n.leaves**
TB	5078	21.33	29	359
H3N2	1829	10.45	26	126
SARS-CoV-2	3707	10.78	47	180
Pertussis	2537	10.85	20	200

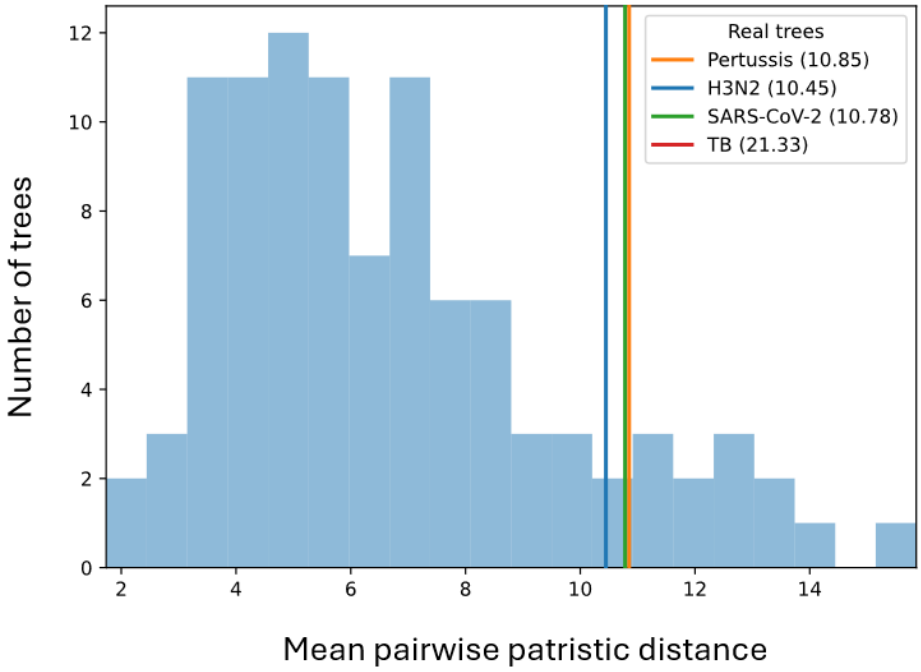


Figure 4.1: Distribution of mean pairwise patristic distances across simulated phylogenies. The histogram shows the distribution of the mean pairwise patristic distance computed over 100 simulated phylogenies generated under the BD-MUT model. The x-axis reports the average pairwise patristic distance for each tree, while the y-axis indicates how many trees fall within each bin. Vertical colored lines mark the corresponding values for the empirical datasets used in this thesis, Pertussis, H3N2, SARS-CoV-2, and TB, with their mean distances reported in parentheses. The empirical trees fall within (or close to) the simulated range of values, indicating that the simulated training data adequately match the evolutionary scale of real pathogen phylogenies.

To further verify this, Figures 4.1 and 4.2 show the distributions of key metrics for simulated trees compared with the empirical values.

Overall, this analysis provides strong evidence that the simulation model produces diverse and biologically plausible phylogenies. This is essential because Phyloscope learns solely from simulated data; therefore, validating the realism of the training distribution increases confidence in the method's applicability to empirical phylogenetic trees.

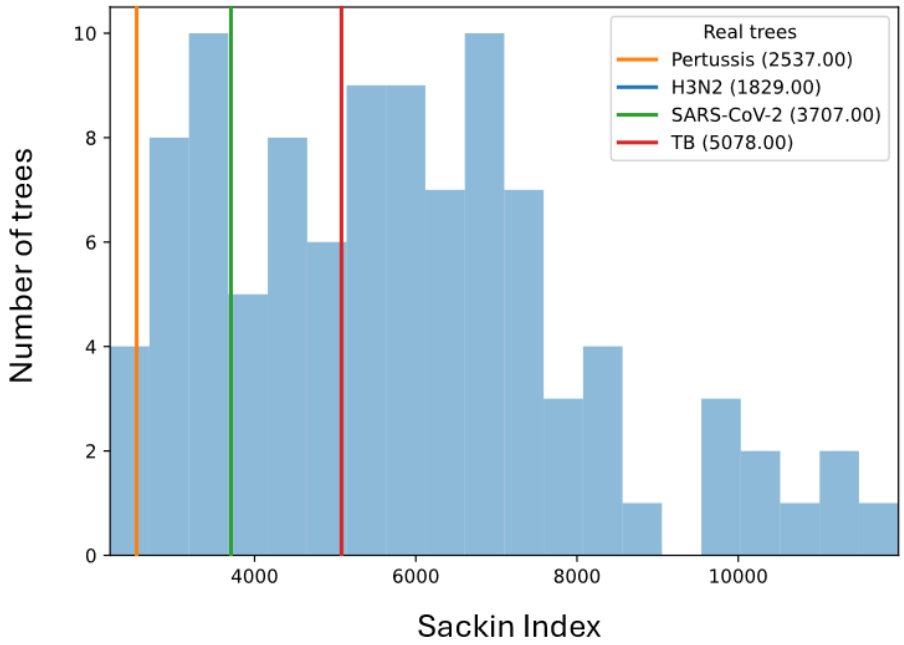


Figure 4.2: Distribution of Sackin index values across simulated phylogenies. The histogram reports the distribution of the Sackin index for 100 simulated phylogenies, a classical measure of tree imbalance where higher values indicate more unbalanced branching structures. The x-axis shows the Sackin index of each simulated tree, while the y-axis represents the number of trees falling into each bin. Vertical colored lines correspond to the Sackin index of the empirical phylogenies used in this thesis, Pertussis, H3N2, SARS-CoV-2, and TB, along with their index values shown in parentheses. The empirical Sackin indices lie within or near the simulated distribution, indicating that the simulated training trees capture a comparable range of topological imbalance to that observed in real pathogen phylogenies.

4.2 Evaluation Metrics

To evaluate the performance of Phyloscope on simulated phylogenies, we monitored both the training and validation loss, which measure how well the encoder learns the embedding space, and standard clustering metrics comparing the predicted clusters to the true labels (after applying the minimum-leaves-per-cluster grouping). Quantitative evaluation is performed only on simulated data, where ground-truth cluster assignments are available.

4.2.1 Contrastive Loss

The model was trained using a smoothed contrastive loss (as described in Chapter 3) which encourages high similarity between embeddings of nodes belonging to the same true cluster and low similarity between embeddings from different clusters. Lower loss values indicate a cleaner separation of evolutionary groups in the latent space.

At convergence (after 20 epochs), the loss values were:

- **Training loss: 0.14**
- **Validation loss: 0.16**

These values show that the encoder successfully learned consistent representations of evolutionary structure across a large set of simulated phylogenies.

4.2.2 Clustering Metrics

To assess the quality of the predicted clusters, we computed three widely used clustering evaluation metrics: ARI, NMI and AMI. All metrics compare the predicted clusters with the ground truth after minimum-leaves-per-cluster grouping. The computation of ARI, NMI, and AMI followed the standard implementations provided in the `scikit-learn` library [25].

Adjusted Rand Index (ARI). ARI measures the agreement between two clusterings based on pairwise assignments. Its values range from:

- 1.0 — perfect agreement,
- 0 — random clustering,
- < 0 — worse than random (rare in practice).

Normalized Mutual Information (NMI). NMI quantifies how much information is shared between two clusterings. Values range from:

- 1.0 — identical clusterings,
- 0 — no mutual information between partitions.

Adjusted Mutual Information (AMI). AMI is a variant of NMI corrected for chance. Its values range from:

- 1.0 — perfect agreement,
- ≈ 0 — no agreement beyond chance.

4.2.3 Clustering Results on Simulated Data

All values below represent the mean ± standard deviation across 100,000 simulated trees used during training and validation (80/20 split), as discussed in the Chapter 3.

Table 4.2: Final clustering metrics on simulated phylogenies.

Metric	Training	Validation
ARI	0.71 ± 0.06	0.66 ± 0.07
NMI	0.74 ± 0.05	0.70 ± 0.06
AMI	0.72 ± 0.05	0.68 ± 0.06

Although empirical phylogenies come with well-established lineage or clade labels, these annotations represent operational ground truth. They are defined through a mixture of historical convention, expert curation, and pathogen-specific heuristics, and they do not necessarily correspond to the fitness shifts that underpin the simulated training data. For this reason, quantitative clustering metrics such as ARI, NMI, and AMI, which rely on a one-to-one correspondence between predicted and true clusters, are not computed on empirical datasets. Instead, empirical results are evaluated qualitatively, in the subsequent sections of this chapter, by comparing the predicted clusters with these operational labels to assess whether the model recovers the major evolutionary groups.

4.2.4 Interpretation

These clustering scores show that:

- the model reliably reconstructs the major mutation-driven clades;
- small clusters are sometimes merged, reducing ARI slightly;
- the gap between training and validation is small, indicating good generalization.

Overall, Phyloscope learns a meaningful and stable embedding space that captures the true evolutionary communities present in the simulated phylogenies.

4.3 Network validation

Threshold selection for clustering. While the contrastive loss shapes the continuous geometry of the embedding space, a discrete cluster assignment is required to visualize the results on empirical phylogenies. For these real-world

datasets, cluster boundaries were obtained by applying a similarity threshold to the node-parent cosine similarity detailed in the previous chapter 3.

The threshold was not fixed a priori, as the appropriate value depends on the empirical distribution of similarities in each tree. Instead, only for inference on empirical data, an *elbow-based heuristic* was used: selecting the threshold at the point where the similarity histogram shows a sharp change in slope. Fig. 4.3 shows an example of the elbow heuristic for clustering. This choice has a purely visual

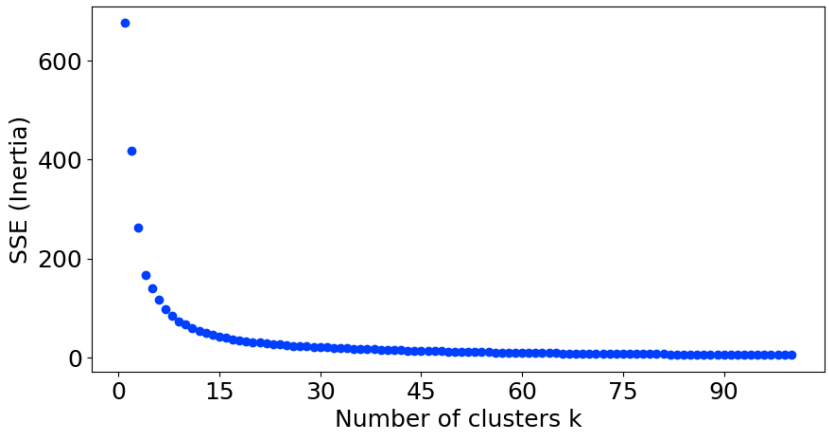


Figure 4.3: Elbow heuristic for selecting the number of clusters in k-means clustering. The plot shows how the **inertia**, the within-cluster sum of squared distances between each point and the centroid of its assigned cluster, changes as a function of the number of clusters k. The x-axis reports the chosen value of k, while the y-axis reports the corresponding inertia. Increasing k always decreases inertia, because clusters become smaller and better fit the data. However, the rate of decrease is not uniform: after an initial steep drop, the curve gradually flattens. The *elbow point* marks the value of k where adding more clusters provides only marginal reductions in inertia. This point offers a practical trade-off between underfitting (too few clusters) and overfitting (too many), and is therefore used as a heuristic to select an appropriate number of clusters.

and interpretative purpose:

- it enables the projection of continuous embeddings onto discrete clusters,
- it produces stable and biologically interpretable clade partitions,
- and it can be easily adjusted by inspecting the histogram for each empirical dataset.

Importantly, this threshold plays *no role* during training or validation on simulated data, where ground-truth cluster labels exist and are used directly. In this

context, the threshold was set to 0.5 for all phylogenetic trees, and this proved to be an optimal choice for visualization purposes for the majority of phylogenies. Its role is limited to the qualitative visualization of cluster structure in real phylogenetic trees, allowing the continuous embedding space to be mapped into discrete evolutionary groups.

Realistically, the threshold is pathogen-specific and has to be fine-tuned by the user, very similarly to Phylowave [9]. Later studies could automate this process.

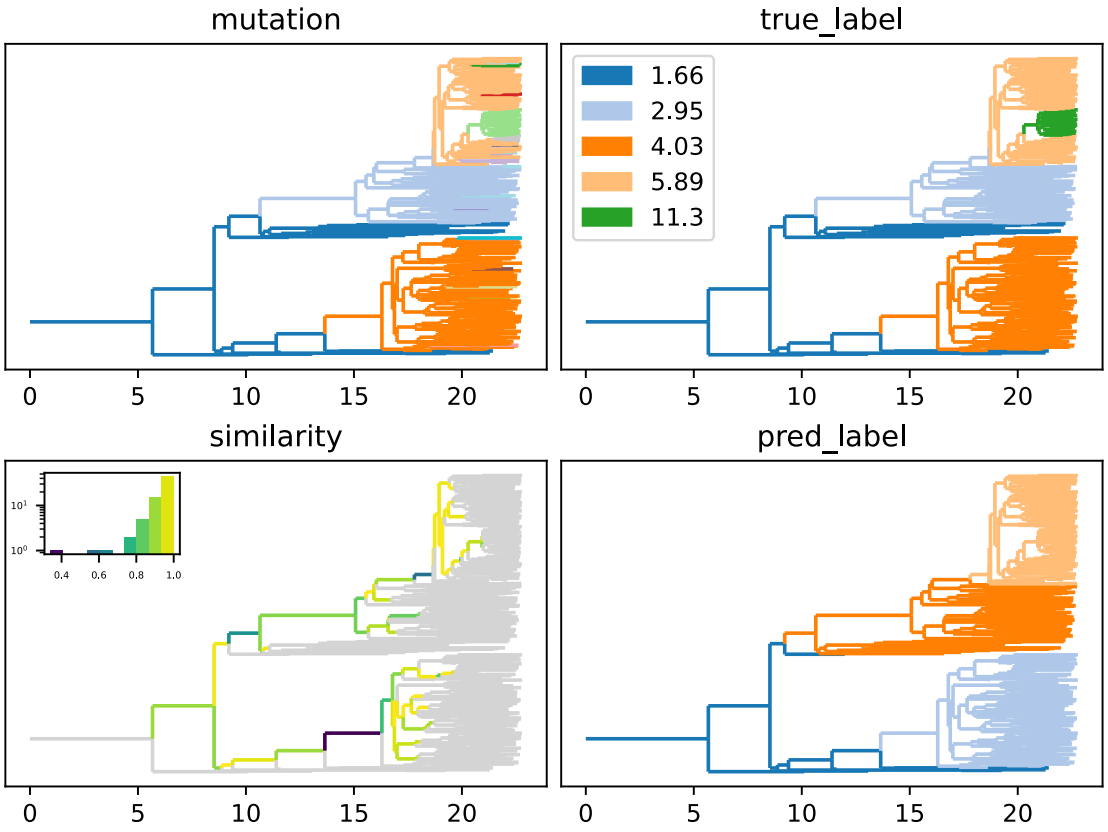


Figure 4.4: Example of validation tree: comparison between mutation values, true cluster labels, embedding-based similarity, and predicted labels for a representative simulated phylogeny. Top: true mutation coloring (left) and corresponding ground-truth cluster assignments with logged birth for each corresponding mutation in the legend (right). Bottom: similarity map derived from the learned node embeddings, with a histogram showing the similarity scores (left), and model-predicted cluster labels (right). The strong correspondence between predicted and true labels indicates that the encoder successfully captures the hierarchical structure of the tree and reconstructs meaningful evolutionary clusters.

Figure 4.4 provides a visual comparison between the ground-truth evolutionary

structure of a simulated phylogeny and the corresponding predictions obtained by applying the prediction head to the embeddings produced by the Phyloscope encoder. The top-left panel shows the true mutation-based coloring of the tree, where all of the simulated mutations are displayed with different colors. The top-right panel illustrates the true cluster labels after the minimum leaves per cluster condition was applied: each color corresponds to a clade originating from a mutation event that satisfies the minimum cluster size criterion. These labels serve as the structural ground truth that the model aims to reconstruct.

The bottom-left panel displays the similarity scores computed from the learned node embeddings. Branches with higher similarity are rendered in more saturated colors, indicating that the encoder successfully captures hierarchical relationships across the tree. The histogram uses cosine similarity values on the x-axis, while the y-axis reports the number of node pairs falling into each similarity bin. To make both very common and very rare similarity values visible, the y-axis is shown on a logarithmic scale. The inset histogram shows two main groups of similarity values, which roughly correspond to pairs of nodes from the same cluster and from different clusters.

Finally, the bottom-right panel shows the predicted labels obtained from the model. The correspondence with the true labels is generally good: major clades are correctly recovered, and the boundaries between predicted clusters closely follow the underlying phylogenetic splits. Comparing predicted results and ground truth, small discrepancies can be found: the green cluster (11.3) is not predicted correctly as it is merged with the light orange one (5.89). The same misclassification occurs for some blue tips (1.66) classified as orange (4.03). Overall, the model recovers the four main clusters, with only minor misclassifications.

This type of four-panel layout is used in all the following figures: the top row always shows the true mutation coloring and the corresponding true cluster labels, while the bottom row displays the similarity map with its histogram and the predicted labels.

Figure 4.5 presents a special validation case in which the simulated phylogeny contains no mutation event large enough to form a valid cluster under the minimum–leaves criterion. As shown in the top row, in the mutation coloring some small clusters are identified, but in the true cluster labels those clusters are not present, since their nodes had less than 17 descendant leaves, considering that the entire tree belongs to a single homogeneous group.

This example is important because it evaluates whether the network introduces artificial structure when none is present. In the bottom-left panel, the similarity map computed from the learned embeddings shows consistently high cosine similarity across almost all branches. The histogram in the inset, displays a narrow peak concentrated at very high similarity values, suggesting very similar node-parent embeddings. This indicates that the encoder correctly interprets the whole tree as

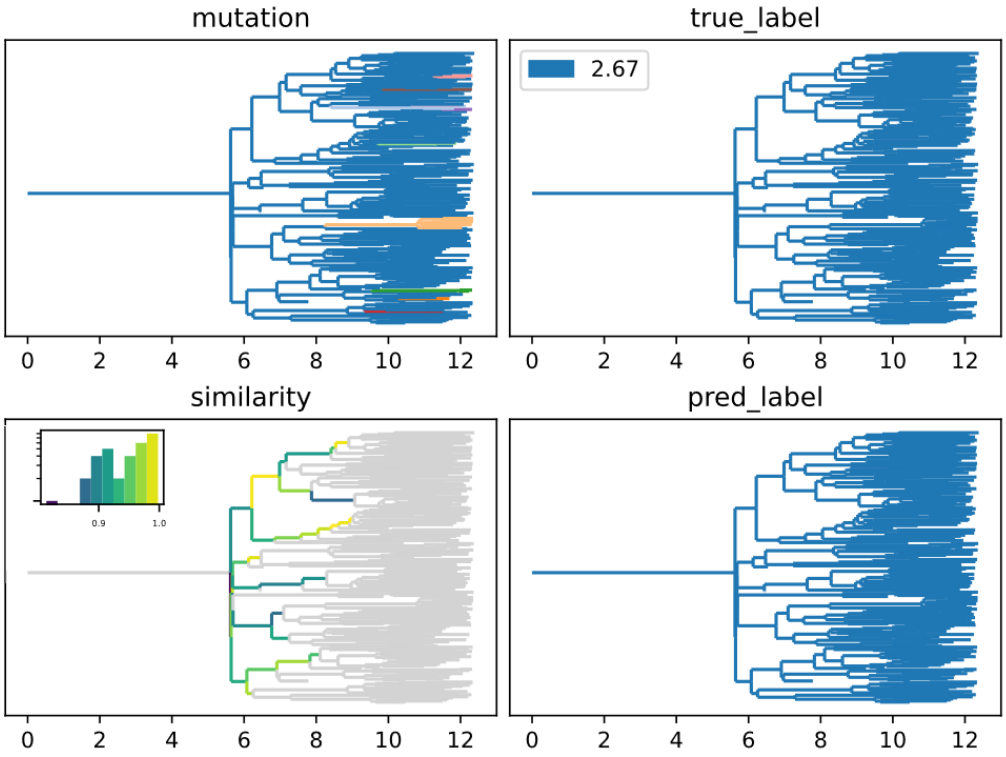


Figure 4.5: Second validation tree example: this phylogeny contains no mutation-derived clusters in the ground truth. Top: true mutation coloring (left), and the corresponding true cluster labels (right), showing a single mutation value across the entire tree, where all nodes belong to the same cluster due to the absence of valid mutation events. Bottom: similarity map derived from the learned node embeddings (left), with a log-scaled histogram of cosine similarity values in the inset, and the predicted labels (right). As expected, the network assigns all nodes to a single cluster.

a single coherent unit, with no internal divisions strong enough to imply separate clusters.

An important consequence of these high similarity scores is that any threshold used to discretize clusters would need to be set very low to artificially create more than one cluster.

The bottom-right panel shows the predicted labels. Phyloscope assigns the same cluster to every leaf, confirming that the model does not create evolutionary structure when the underlying data does not support it. This conservative behavior is essential: the method avoids false positives and only produces clusters when the

learned embeddings exhibit meaningful separation.

Overall, this case demonstrates that the encoder behaves reliably in the absence of true diversification and produces embeddings that remain stable and biologically consistent.

4.4 Inference on Empirical data

From validation trees to empirical ones Having illustrated two representative validation examples, one with clear mutation-driven structure and one with no detectable clusters, the behavior of the model on simulated data has been illustrated. The analysis can therefore proceed to the inference results on empirical phylogenetic datasets.

Table 4.3: Phyloscope similarity thresholds selected for each empirical dataset.

Dataset	Threshold
Pertussis	0.71
SARS-CoV-2	0.85
TB	0.80
H3N2	0.90

Table 4.3 summarizes the optimal thresholds found for each pathogen. The similarity threshold was selected separately for each dataset by visually inspecting the histogram of cosine similarities and choosing the point corresponding to the elbow of the distribution. This threshold is used only to convert the continuous embedding space into discrete clusters for visualization and comparison with ground-truth clade labels. The minimum leaves per cluster threshold was set to 17 in all experiments, matching the value used during training and validation. This choice provided a good balance between avoiding very small, unstable clusters and preserving the main mutation–derived clades. An exception was made for the H3N2 dataset: due to the presence of many small clades terminating in several closely related leaves, the threshold was increased to 25 to prevent the formation of many small clusters.

In the case of H3N2 (Figure 4.6), Phyloscope is able to reconstruct the main structure of the phylogeny, capturing most of the large clades. The true clade assignments (top-right panel) contain many closely related sublineages.

The predicted labels (bottom-right panel) follow these groups reasonably well. The deepest splits of the tree are reproduced correctly, and the main backbone of the phylogeny is preserved. Several of the larger clusters, in particular `2a`, `2a.1`, and `2a.3`, are recovered as coherent units.

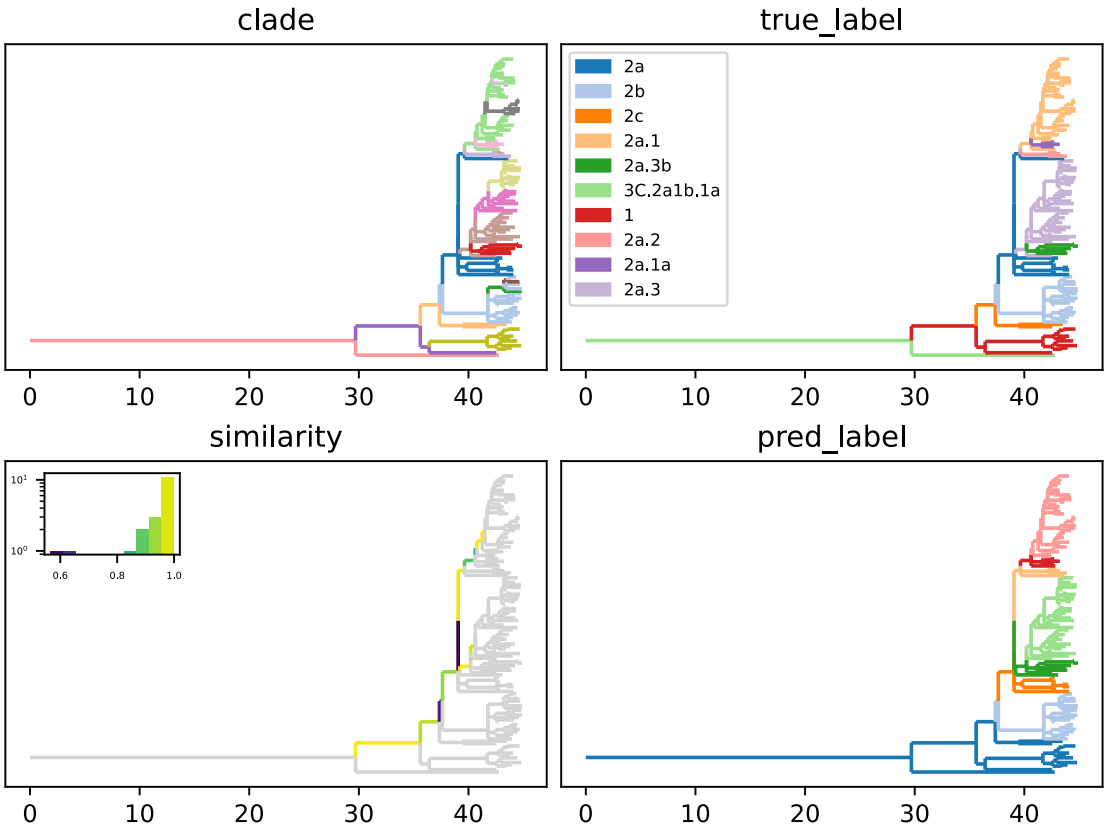


Figure 4.6: Inference results for the H3N2 phylogeny (subsampled from 2022). Top row: the true clade assignments derived from metadata (right) and the corresponding mutation-based coloring (left), displaying the numerous closely related sublineages that characterize recent H3N2 seasonal evolution. Bottom row: the similarity map computed from Phyloscope's learned embeddings (left), with a log-scaled histogram of cosine similarities in the inset, and the predicted cluster labels produced by the model (right). Phyloscope successfully recovers the major circulating clades, including the prominent groups `2a`, `2a.1`, and `2a.3`, and preserves the deeper structural backbone of the tree. Because this tree contains many small, shallow sublineages, the minimum cluster-size threshold was increased to 25 tips to avoid fragmentation into numerous tiny clusters. Despite merging some of the closest sublineages, the model provides a coherent reconstruction of the main evolutionary groups present in the dataset.

Some of the finer subclades are merged by the model, such as `2a.3b` and `2a.3` or `2a` and `2a.2`, and reflecting the difficulty of separating very similar H3N2 clades.

The similarity map (bottom-left) shows a clear block corresponding to the

main H3N2 backbone, while higher similarities are concentrated within each major seasonal group. This justified using a slightly larger minimum cluster size (`min_tips_per_cluster` = 25) for this dataset, preventing the formation of numerous tiny clusters and stabilizing the predicted structure.

Overall, Phyloscope provides a coherent reconstruction of the H3N2 phylogeny: the main circulating clades are recovered, the major evolutionary splits are maintained, and only finer sublineages are occasionally merged.

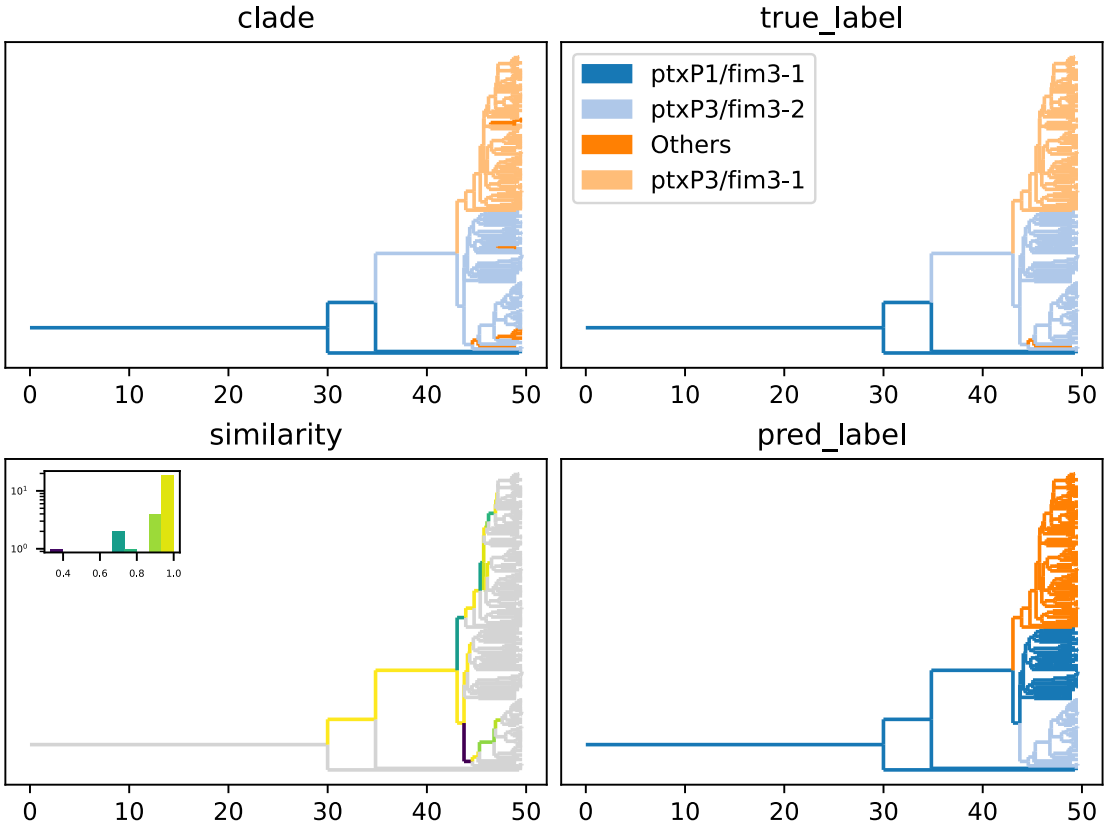


Figure 4.7: Inference results for the Pertussis phylogeny (2009–2011 subsample). Top row: true clade assignments derived from metadata (right) and the corresponding mutation-based coloring (left) after minimum leaves per cluster grouping. Bottom row: similarity map computed from the learned node embeddings (left), including a log-scaled histogram of cosine similarities in the inset, and the predicted cluster labels returned by Phyloscope (right). Two of the three major clades (`ptxP1/fim3-1` and `ptxP3/fim3-2`) are successfully recovered, with predicted clusters that match the true structure of the tree.

Figure 4.7 shows the inference results for the Bordetella pertussis phylogeny (2009–2011 subsample). This dataset contains only three major clades of substantial

size, as the `Others` category is represented by very few tips and therefore falls below the minimum cluster-size threshold. Increasing the minimum leaves per cluster hyperparameter would have omitted the `Others` cluster. However, other relevant clusters would not have been considered.

One true clade is not identified: instead, the model detects an additional cluster with high internal similarity, a behavior also observed in PhyloWave for this dataset. Because the `Others` clade contains too few tips, it falls below the minimum cluster-size threshold and is therefore not treated as a separate cluster.

Overall, the model reconstructs the main evolutionary relationships in the Pertussis phylogeny, detecting one additional undetected cluster.

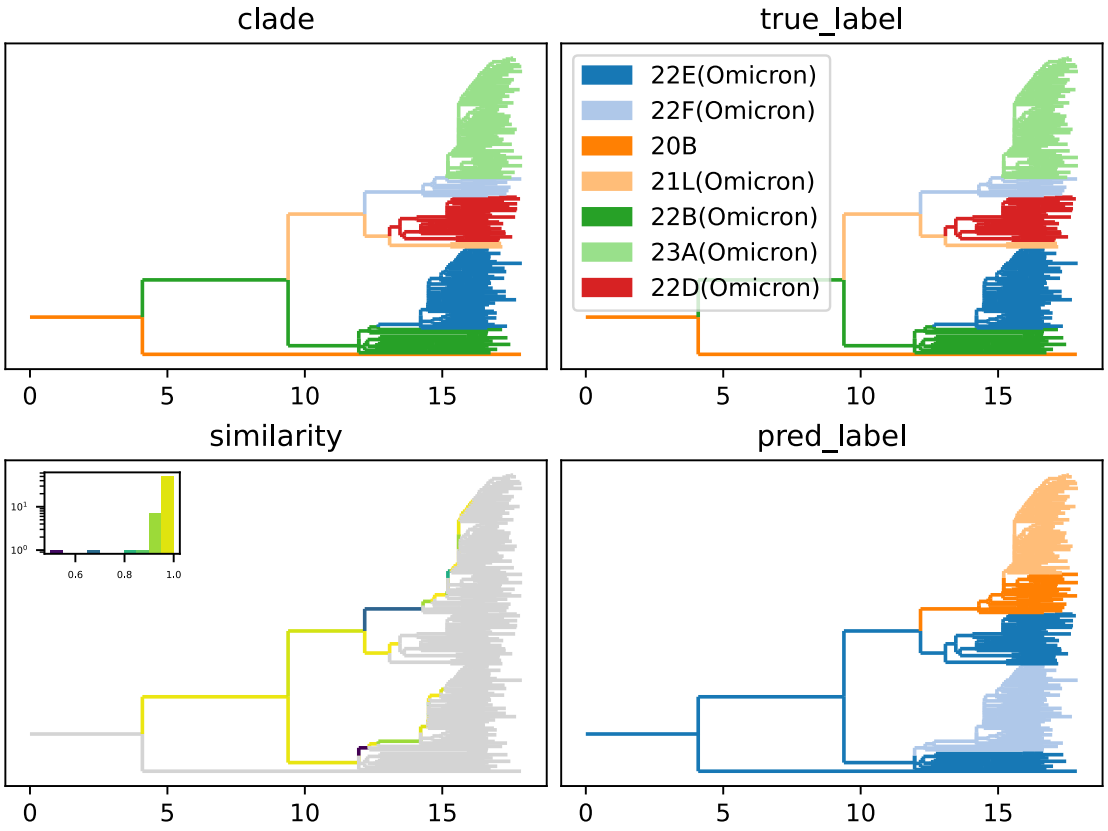


Figure 4.8: Inference results for the SARS-CoV-2 phylogeny (subsampled from 2023). Top row: mutation-based coloring (left) and true clade assignments (right), showing the dominant Omicron sublineages present in the dataset. Bottom row: similarity map derived from the learned node embeddings (left), including a log-scaled histogram of cosine similarities in the inset, and the predicted cluster labels obtained from Phyloscope (right).

Figure 4.8 shows the inference results for the SARS-CoV-2 phylogeny (subsampled from 2023). This dataset consists almost entirely of Omicron sublineages, many of which differ by only a few mutations, making their separation particularly challenging for any similarity-based model.

Phyloscope successfully recovers the major Omicron groups, producing a predicted structure that broadly follows the true clade assignments. The large-scale splits are captured accurately, and the global arrangement of the tree is consistent with the true evolutionary relationships. However, several of the finer sublineages (e.g. 22F, 23A and 22D, 21L) exhibit very small genetic distances and are therefore merged by the model into larger predicted clusters.

Despite these limitations, the essential backbone of the SARS-CoV-2 tree is preserved: the main Omicron lineages are clearly distinguished, and the predicted clusters remain biologically coherent. This behavior is expected given the limited genetic variation present across recent Omicron subclades and is consistent with observations made in similar frameworks such as PhyloWave.

Figure 4.9 shows the inference results for the Tuberculosis (TB) phylogeny. Unlike the viral datasets, this tree contains only a few broad clades, all belonging to the `Euro-American` lineage, but with several well-defined subgroups such as `Euro_American_LAM`, `Euro_American_Harleem`, and `Euro_American_Ural`.

Phyloscope correctly recovers the major `Euro-American` backbone of the tree: the large top and bottom clusters are reconstructed consistently and match the true labels, which is a non-trivial aspect of this dataset due to the long branches separating these groups. However, the model has more difficulty distinguishing among the internal Euro-American sublineages. In particular, `Euro_American_LAM` is split into two predicted clusters, and several tips belonging to `Euro_American_Ural` are misclassified. Furthermore, the `Euro_American_Harleem` clade is not recovered as a distinct group, instead it is merged with the bigger `Euro-American`

Despite these limitations, the global tree structure is largely preserved, and the main Euro-American division is captured correctly. This outcome is consistent with the behavior observed in other empirical datasets: the model performs well on broad-scale splits, while finer distinctions between closely related sublineages are more challenging to resolve based solely on embedding similarity.

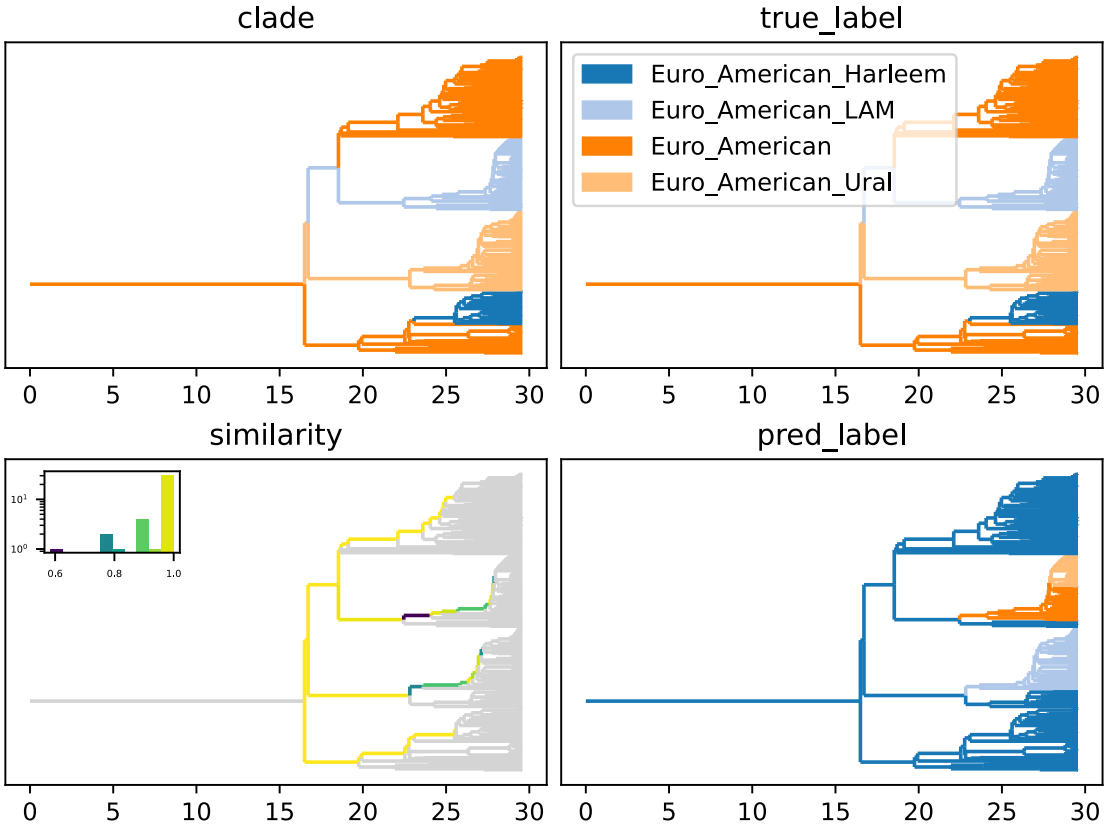


Figure 4.9: Inference results for the Mycobacterium tuberculosis (TB) phylogeny. Top row: mutation-based coloring (left) and true clade assignments (right), showing the major Euro-American sublineages present in the dataset. Bottom row: similarity map produced by the learned node embeddings (left), including a log-scaled histogram of cosine similarities, and the resulting predicted cluster labels from Phyloscope (right).

Chapter 5

Discussion & Conclusions

5.1 Discussion

In this thesis, the community detection problem in phylogenetic trees is addressed, which is crucial for the identification of clades with increased fitness that could create threats for public health (for example escaping immunity).

This thesis introduced *Phyloscope*, a new framework for detecting clades in phylogenetic trees using recursive neural networks and supervised contrastive learning. This work brought together two core contributions: a recursive architecture (hybrid CBLV–MLP) for producing the encodings of the phylogenetic tree and a smoothed contrastive loss formulation that allowed the embeddings to reflect the mutation dynamics.

To train the network, a large dataset of phylogenetic trees was simulated under a birth-death model extended with mutation events. Once simulated trees were preprocessed and converted into structured samples, the recursive encoder was trained to summarize node information into a fixed-dimensional encoding. A contrastive loss led pairs of nodes with similar birth-rate to be close in embedding space, while pushing apart pairs with different anchors. The more the respective birth rates were different, the more their encodings were pushed apart.

The final output of Phyloscope is a matrix of cosine similarities between the nodes of a given tree. By applying a similarity threshold, this continuous representation is converted into discrete clusters. A key advantage of this threshold is interpretability: it can be easily fine-tuned by looking at the similarity histogram. In empirical applications, a simple elbow technique was applied to select the optimal number of clusters. This selection proves to be more intuitive than previous approaches, such as PhyloWave [9], where a different threshold had to be fine-tuned by experts.

Across both simulated and empirical datasets, Phyloscope was able to correctly detect the majority of clades. In validation trees, the network logically identified

clades with respect to their birth rates, also proving not to hallucinate when no mutations were present. On the four empirical phylogenies, (H3N2, Pertussis, SARS-COV-2 and Tuberculosis), the network reproduced the major lineage splits and occasionally merging very similar clusters. In the Pertussis tree, an additional lineage was detected, while for the Tuberculosis tree some misclassification problems were shown.

The empirical analyses highlighted some limitations. It is unclear how the performance of Phyloscope would generalize for trees larger than those that appeared in the training data. However, given the recursion, it seems plausible it would work well within trees up to 500 tips (which was the upper bound for the simulated trees).

Overall, Phyloscope:

- suggests a novel way of *automatically detecting fitness-increasing mutation events* in pathogen phylogenies,
- offers a *real-time, scalable inference* that does not rely on expert-defined metrics,
- *generalizes* reasonably to *real-world pathogen* phylogenies.

Importantly, using simulated phylogenies is not a limitation of the method. The simulation framework can be made more realistic very easily, and Phyloscope would immediately benefit from this increased realism. This is a clear advantage over analytical approaches such as Phylowave, where extending the underlying mathematical model is far more difficult. Therefore, the ability to plug in richer and more accurate simulations should be viewed as one of the strengths of the framework.

5.2 Conclusions and Future Work

This thesis introduced Phyloscope, a novel framework for clade detection in phylogenies based on recursive neural encodings and contrastive learning. The method demonstrates that learned representations can capture evolutionary structure in a way that is competitive with existing approaches while offering greater flexibility and conceptual simplicity. Across simulated and real-world datasets, the model was able to reconstruct major clades, highlight regions of high similarity in the embedding space, and generalize across pathogens with distinct evolutionary profiles.

Several promising directions for future work emerge from this study. First, the current loss function could be extended or replaced by a more expressive formulation that better captures the hierarchical nature of phylogenies. Second, the

model architecture itself could be enriched: while a simple MLP proved effective, more powerful designs may further improve encoding quality. Another promising direction for future work concerns the simulation framework. Although the normal distribution used for the birth-rate scaling factor makes large fitness decreases extremely unlikely (occurring only several standard deviations away from the mean), the framework itself is flexible and can easily accommodate more complex evolutionary scenarios. Future extensions could therefore incorporate mutations with both increasing and decreasing fitness, or adopt more expressive models such as BDEI or BDSS, ultimately enabling the model to train on larger and more realistic phylogenies. Training on larger phylogenies remains a major goal: with appropriately designed simulations, Phyloscope could eventually operate on full pathogen trees without relying on subsampling, enabling direct comparisons with state-of-the-art methods like Phylowave.

An additional research direction concerns threshold selection. A principled, data-driven approach for determining pathogen-specific thresholds, possibly leveraging the similarity histogram or embedding geometry, would significantly enhance the usability and automation of the method.

Finally, extending the model with an inference head capable of estimating birth rates or other evolutionary parameters would open the door to true phylodynamic inference within the Phyloscope framework.

Overall, this work establishes a foundation for learning-based clade detection and points to several exciting opportunities for extending Phyloscope toward a more robust, scalable, and biologically grounded tool for phylogenetic analysis.

Bibliography

[1] Jennifer L. Gardy and Nicholas J. Loman. «Towards a genomics-informed, real-time, global pathogen surveillance system». In: *Nature Reviews Genetics* 19.1 (2018), pp. 9–20. DOI: 10.1038/nrg.2017.88. URL: https://www.nature.com/articles/nrg.2017.88 (cit. on p. 1).

[2] W. Tanner Porter, David M. Engelthaler, and Crystal M. Hepp. «Genomic Epidemiology for Estimating Pathogen Burden in a Population - Volume 31, Supplement—April 2025 - Emerging Infectious Diseases journal - CDC». en-us. In: (Apr. 2025). DOI: 10.3201/eid3113.241203. URL: https://wwwnc.cdc.gov/eid/article/31/13/24-1203_article (cit. on pp. 1, 17, 18).

[3] Claudia Alteri et al. «Genomic epidemiology of SARS-CoV-2 reveals multiple lineages and early spread of SARS-CoV-2 infections in Lombardy, Italy». en. In: *Nature Communications* 12.1 (Jan. 2021). Publisher: Nature Publishing Group, p. 434. ISSN: 2041-1723. DOI: 10.1038/s41467-020-20688-x. URL: https://www.nature.com/articles/s41467-020-20688-x (cit. on pp. 1, 17, 25, 39).

[4] Philippe Lemey, Andrew Rambaut, Alexei J. Drummond, and Marc A. Suchard. «Bayesian phylogeography finds its roots». In: *PLoS Computational Biology* 5.9 (2009), e1000520. DOI: 10.1371/journal.pcbi.1000520 (cit. on pp. 1, 2).

[5] Joseph Felsenstein. *Inferring Phylogenies*. Sunderland, MA: Sinauer Associates, 2004. ISBN: 9780878931774 (cit. on pp. 1, 3).

[6] Oliver G. Pybus and Andrew Rambaut. «Evolutionary analysis of the dynamics of viral infectious disease». In: *Nature Reviews Genetics* 10.8 (2009), pp. 540–550. DOI: 10.1038/nrg2583. URL: https://www.nature.com/articles/nrg2583 (cit. on p. 2).

[7] Tanja Stadler. «Sampling-through-time in birth–death trees». In: *Journal of Theoretical Biology* 267.3 (2010), pp. 396–404. DOI: 10.1016/j.jtbi.2010.09.010 (cit. on pp. 2, 3).

[8] Stephen W. Attwood, Sarah C. Hill, David M. Aanensen, Thomas R. Connor, and Oliver G. Pybus. «Phylogenetic and phylodynamic approaches to understanding and combating the early SARS-CoV-2 pandemic». en. In: *Nature Reviews Genetics* 23.9 (Sept. 2022). Publisher: Nature Publishing Group, pp. 547–562. ISSN: 1471-0064. DOI: 10.1038/s41576-022-00483-8. URL: https://www.nature.com/articles/s41576-022-00483-8 (cit. on pp. 2, 17, 18, 20, 25).

[9] Noémie Lefrancq, Loréna Duret, Valérie Bouchez, Sylvain Brisse, Julian Parkhill, and Henrik Salje. «Learning the fitness dynamics of pathogens from phylogenies». In: *Nature* 637 (2025), pp. 683–690. DOI: 10.1038/s41586-024-08309-9. URL: https://www.nature.com/articles/s41586-024-08309-9 (cit. on pp. 2, 3, 18, 22, 26, 30, 47, 53, 61, 70).

[10] J. Voznica, A. Zhukova, V. Boskova, E. Saulnier, F. Lemoine, M. Moslonka-Lefebvre, and O. Gascuel. «Deep learning from phylogenies to uncover the epidemiological dynamics of outbreaks». en. In: *Nature Communications* 13.1 (July 2022). Publisher: Nature Publishing Group, p. 3896. ISSN: 2041-1723. DOI: 10.1038/s41467-022-31511-0. URL: https://www.nature.com/articles/s41467-022-31511-0 (cit. on pp. 2, 15, 18–20, 23, 27, 28, 44).

[11] Ismaël Lajaaiti, Sophia Lambert, Jakub Voznica, Hélène Morlon, and Florian Hartig. *A Comparison of Deep Learning Architectures for Inferring Parameters of Diversification Models from Extant Phylogenies*. en. Mar. 2023. DOI: 10.1101/2023.03.03.530992. URL: http://biorxiv.org/lookup/doi/10.1101/2023.03.03.530992 (cit. on pp. 2, 3, 18–21).

[12] Simon Y. W. Ho and Sebastián Duchêne. «Molecular-clock methods for estimating evolutionary rates and timescales». In: *Molecular Ecology* 23.24 (2014), pp. 5947–5965. DOI: 10.1111/mec.12953. URL: https://onlinelibrary.wiley.com/doi/10.1111/mec.12953 (cit. on p. 3).

[13] Vladimir N. Minin and Marc A. Suchard. «Fast, Accurate and Nonparametric Estimation of Population Size Dynamics Using ESP-Based Methods». In: *Molecular Biology and Evolution* 25.7 (2008), pp. 1459–1471. DOI: 10.1093/molbev/msn090. URL: https://academic.oup.com/mbe/article/25/7/1459/1124132 (cit. on p. 3).

[14] P. Frasconi, M. Gori, and A. Sperduti. «A general framework for adaptive processing of data structures». In: *IEEE Transactions on Neural Networks* 9.5 (Sept. 1998), pp. 768–786. ISSN: 1941-0093. DOI: 10.1109/72.712151. URL: https://ieeexplore.ieee.org/document/712151 (visited on 11/22/2025) (cit. on pp. 3, 18, 19, 22).

[15] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. *Representation Learning with Contrastive Predictive Coding*. arXiv:1807.03748 [cs]. Jan. 2019. DOI: 10.48550/arXiv.1807.03748. URL: http://arxiv.org/abs/1807.03748 (cit. on pp. 3, 18, 23).

[16] Luke J. Harmon. *Introduction to birth-death models · Phylogenetic Comparative Methods*. URL: https://lukejharmon.github.io/pcm/chapter10_birthdeath/ (cit. on pp. 5, 7–10, 13).

[17] Jeanne Lemant et al. *Robust, Universal Tree Balance Indices | Systematic Biology | Oxford Academic*. URL: https://academic.oup.com/sysbio/article/71/5/1210/6567363?login=false (cit. on pp. 11, 12).

[18] *treesimulator/README.md at main · evolbioinfo/treesimulator*. URL: https://github.com/evolbioinfo/treesimulator/blob/main/README.md (cit. on p. 13).

[19] Di Jin, Zhizhi Yu, Pengfei Jiao, Shirui Pan, Dongxiao He, Jia Wu, Philip S. Yu, and Weixiong Zhang. *A Survey of Community Detection Approaches: From Statistical Modeling to Deep Learning*. arXiv:2101.01669 [cs]. Aug. 2021. DOI: 10.48550/arXiv.2101.01669. URL: http://arxiv.org/abs/2101.01669 (cit. on pp. 16, 17, 21, 23).

[20] Xing Su et al. «A Comprehensive Survey on Community Detection with Deep Learning». In: *IEEE Transactions on Neural Networks and Learning Systems* 35.4 (Apr. 2024). arXiv:2105.12584 [cs], pp. 4682–4702. ISSN: 2162-237X, 2162-2388. DOI: 10.1109/TNNLS.2021.3137396. URL: http://arxiv.org/abs/2105.12584 (cit. on pp. 16, 21, 23).

[21] Leo A Featherstone, Joshua M Zhang, Timothy G Vaughan, and Sebastian Duchene. «Epidemiological inference from pathogen genomes: A review of phylodynamic models and applications». In: *Virus Evolution* 8.1 (June 2022), veac045. ISSN: 2057-1577. DOI: 10.1093/ve/veac045. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC9241095/ (cit. on pp. 17–19, 23, 25, 27).

[22] Matthew Sainsbury-Dale, Andrew Zammit-Mangion, and Raphaël Huser. «Likelihood-Free Parameter Estimation with Neural Bayes Estimators». In: *The American Statistician* 78.1 (2023), pp. 1–14. DOI: 10.1080/00031305.2023.2249522. URL: https://doi.org/10.1080/00031305.2023.2249522 (cit. on pp. 19, 20, 27).

[23] Chaoyue Sun et al. *DeepDynaForecast: Phylogenetic-informed graph deep learning for epidemic transmission dynamic prediction | PLOS Computational Biology*. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1011351 (cit. on p. 22).

[24] Erik M. Volz and Xavier Didelot. «Quantifying lineage fitness from phylogenies». In: 23.1 (2025), e3002567. URL: https://www.biorxiv.org/content/10.1101/2025.09.29.679185v1 (cit. on pp. 26, 27).

[25] Fabian Pedregosa et al. «Scikit-learn: Machine Learning in Python». In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on p. 58).