Computational Stemmatology Literature Review

Computational Stemmatology Literature Review

November 2023

Student: Darren Zammit s5284236

Contents

List of Tables

List of Figures

1 A Brief Introduction to Computational Stemmatology

Copying a text multiple times over long periods of time usually results in changes in its content. Stemmatology [1] is a methodical approach to textual criticism and is based on the assumption that if two or more surviving witnesses have common errors then they were likely derived from a common intermediate ancestor (hyparchetype). Relations between the lost ancestors are determined similarly and such predicted ancestors are placed in with all surviving witnesses in a family tree or ”stemma” descended from a single (not necessarily known) ”archetype”.

Once the stemma is derived, the text of the archetype is constructed by examining variants from the hyparchetypes closest to the archetype and selecting the most likely ones. This phase is called selection. The most common variant at each locus (position in the script) at the same level of the tree, is selected as the most likely variant. If multiple variants occur equally often, then the philologist must use his judgement to determine which makes the most sense. A common example is that scribes tend to replace more difficult words by simpler words, thus whilst constructing the archetype, the philologist typically would favour the more complex and sometimes older words that became less often used over time.

The text must then undergo the process of ”examination” during which the philologist must identify errors as in some loci it may be that none of the surviving variants preserve the original text. If the text is determined to be corrupt, it must be corrected through ”emendation”. At the end of this process it is sometimes possible to reconstruct a text closer to the original than any of the surviving witnesses.

The process of deriving a stemma has been studies in mathematical terms [1], however some of the crucial steps in the process are largely impregnable to algorithmic analysis. Usually, familiarising oneself with a text and its contexts is instrumental in understanding its transmission which can be very time-consuming since the philologist often starts knowing very little about the original text or its author(s), their writing style, the environment where the text was written or even the time-period it was written in. This information plays a crucial role in determining the direction of copying and thus finding the root of the stemma. With each successive analysis of the scripts, the philologist identifies more subtleties. Manual stemmatology can still lead to the philologist to find patterns which do not exist since every collection of manuscripts is unique and may follow very different and complicated patterns which humans are typically not well suited to fully understand. This non-linearity makes the process difficult to program fully, however some steps in the process have been computerised to save time for the philologist or derive a hypothetical stemma when one deals with impregnable or very numerous scripts.

There exists a wide variety of approaches in dealing with textual and stemmatic reconstruction and an equally wide variety of tools are available [?]. Computer tools have simplified many steps in text editing. Moreover, computer simulations allow for the study of the evolution of large volumes of textual data. On top of this, computational stemmatology has potential applications in plagiarism detection [10], analysing the evolution of computer viruses [11] and content-based social network analysis [14] since any algorithm used in computational stemmatology in some way relies on comparing scripts.

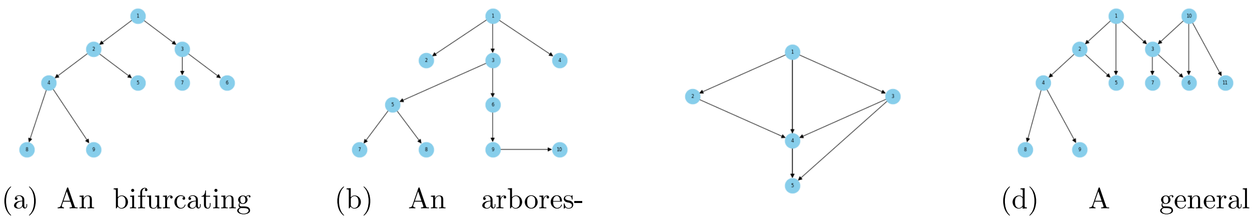
Graphical modelling takes a central role in constructing stemmata [1]. Contemporary methods – which are primarily analogous to those in phylogenetics, the study of the evolutionary history of species – can be split into distance-based [2], parsimony-based [3], and statistical methods [4]. In these methods every word in the text is equally important. In reality there are many different kinds of errors and changes, some more significant than others, leading to the algorithm failing to faithfully construct a stemma. For example, spelling mistakes or changing symbols should have far less impact on a script than adding or redacting text. Another issue which these methods struggle to address is that of contamination, that is when a text is copied from multiple manuscripts [1]. If different scripts were used for different chapters then this could lead to outputs which faithfully reflect the stemma of most chapters but not all if different chapters were copied from different manuscripts. Phylogenetic methods are usually only capable of generating bifurcating trees which are incapable of explaining contaminated stemmata.

Since Lee's seminal paper [5] in applying computational phylogenetics models in stemmatology some advances in computational stemmatology have been made. However, most papers in computational stemmatology are authored by computer scientists and philologists with mathematicians and statisticians representing a small minority of authors [1]. Moreover, algorithms designed specifically for stemmatology, even ones whose framework is adapted for stemmatolology, are very few in number [1]. Computational methods usually rely on transforming the data into analogues of genetic data and feeding it directly into a phylogenetics program as though it were phylogenetic data, with very few algorithms developed exclusively for stemmatology.

One area of computational stemmatology which has not been very well studied is the rooting of a stemma. So far most of the work done in computational stemmatology has been in generating the stemma however the rooting is almost always done manually. This is due to the fact that whilst rooting a stemma, the philologist relies on context such as the historical era, the age of words, changes in spelling etc. which computerised methods are incapable of identifying without highly detailed datasets about such changes.

2 Graph Theory in Stemmatology and Phylogenetics

Approaches to formalising stemmata rely on graph theory as a framework. A graph G is a pair of a set of vertices $\mathcal{V}$ and a set of edges conntecting the vertices $\mathcal{E}$. Vertices represent entities, in the case of stemmata the variants and in phylogenetics the species, whereas edges represent relationships between the entities. A graph said to be undirected if it is composed of a set of vertices and a set of undirected edges (unordered pairs of vertices), and directed if it is composed of a set of nodes and a set of directed edges (ordered pairs of vertices). The two vertices forming an edge are said to be the endpoints of that edge. An edge connecting vertices A and B can be denoted (A, B), in which case A and B are said to be adjacent. The neighbourhood of vertex A is the subgraph formed by all of its adjacent vertices. The degree of a vertex is the number of edges connecting to it. A vertex is called a leaf if it has degree 1. A graph is said to be connected if there exists a path (a sequence of edges which joins a sequence of vertices) between any two vertices. A cycle is a path of vertices and edges in which a vertex is both the start and the end of the path. A directed acyclic graph (DAG) is a directed graph with no directed cycles consisting of vertices and edges with each edge directed from one vertex to another in such a manner that it is impossible to start at any vertex and follow any sequence of edges looping back to the starting vertex. A tree is an undirected graph in which any pair of vertices are connected by only one path. A polytree is a DAG whose undirected graph is a tree. In the absence of contamination, the stemma usually takes the form of a polytree [1]. Trees are used to describe the genesis of related objects that evolve without interfering with one another. In biology, the leaves are usually extant species or groups of thereof, whereas in stemmatology the leaves represent extant witnesses of a text. Most current phylogenetic models produce strictly bifurcatinging or "binary" trees wherein each vertex has at most two descendants.



(a) An bifurcating tree.

(b) An arborescence.

(c) A DAG.

(d) A general stemma.

Figure 1: The main types of graphs in stemmatology and phylogenetics. (d) is a general stemma with more than one root, contamination and more than two children per node.

A rooted tree is a tree in which one vertex is taken as the root. Mathematically, this designation is arbitrary since the root can occur anywhere in the tree. The edges of such a tree can be assigned an orientation, either all towards or all away from the root, in which case the structure becomes a directed rooted tree. An arborescence is a directed rooted tree with one unique directed path to any vertex from the root. In phylogenetics, unrooted trees or "networks" only display the relatedness of the leaves and do not represent a hypothesis of ancestry. To transform the network into a rooted tree, one must determine which changes are more recent than others. A rooted phylogenetic tree is a directed tree with a unique vertex representing the most recent common ancestor of all the taxa represented by the leaves of the tree.

The most common method for rooting phylogenetic trees is by using an outgroup [1], that is, a taxon or set of taxa which is a relative of the group of taxa under study (ingroup) which is related to the ingroup but is lacks a sufficient number of characteristics common amongst the ingroup. The outgroup should be similar enough to allow inference, yet different enough to be distinguished from the ingroup. The ingroup members should be more closely related to one another than to the outgroup. Points were an outgroup is connected to the rest of the tree is taken as the root. Analogues to phylogenetic outgroups are sometimes found in traditions which incorporate texts or parts of texts from other traditions or early translations. Unfortunately for stemmatologists, stemmatological outgroups are rare and thus they must determine the root via other methods. Reticulation is when two taxa merge and create a new one and is largely analogous to hybridisation, where two species interbreed or in horizontal gene transfer, where genes migrate from one species to another (such as from viruses to bacteria). Contamination is analogous to reticulation and can only be explained via networks rather than trees and it may even necessitate multiple roots rather than just one such as when multiple variants are published at once or when manuscripts which include the root have been lost.

In phylogenetics trees the taxa are placed as leaves at the end of the tree and are separated by internal nodes which correspond to the common ancestors of the taxa. The number of internal nodes in a phylogenetic tree depends on the number of taxa in the tree and whether the type of tree. The relationship between the number of taxa n and the number of internal nodes i in a binary phylogenetic tree is $i = n - 1$ since each taxon contributes one leaf node and each internal node has two child nodes however the root node does not have an incoming edge, thus it is not counted as an internal node. In 2 the inner nodes are the points separating the taxa. This is also an example of an ultrametric tree where the taxa ara all equidistant from the root.

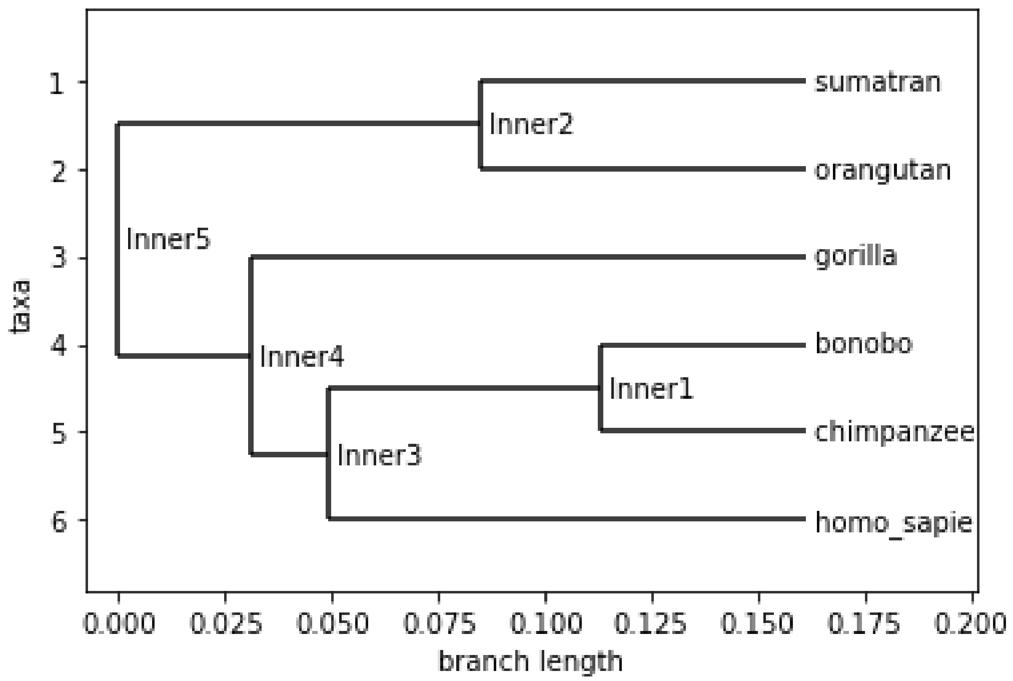


Figure 2: A phylogenetic tree of some primates.

3 Generating a Stemma's structure

3.1 Distance Based Methods

Distance based methods are two-step processes. In the first step one calculates a symmetric dissimilarity matrix with zeros on the diagonal for the input whilst the second step involves constructing a tree from the distance matrix. Such matrices do not generally satisfy the triangle inequality and thus are referred to dissimilarity matrices rather than distance matrices. Many distance-based tree estimation methods are polynomial time and efficient, and thus are favoured over most other algorithms which tend to be computationally expensive [**?**]. As an input distance matrix methods require a collation and produces as output a pairwise distance matrix.

Definition 3.1. *An ultrametric matrix is an $n \times n$ matrix M corresponding to distances between the leaves in a rooted edge-weighted tree T (with non-negative edge weights) where the sum of the edge weights in the path from the root to any leaf of T does not depend on the selected leaf.*

3.2 Distance Measures for Text

Distance metrics are the most common methods used for measuring the similarity between words. In natural language processing such techniques are often used for spell-checking, auto-completing sentences or correcting words and sentences. There exist plenty of distance metrics which can be used to compute and measure similarities between texts. Apart from Stemmaweb [19], there exists no stemmatological tool which allows the conversion of a collation to a distance matrix or to pseudo-DNA, which would be required to easily produce distance matrices or trees with bioinformatic software [1].

Definition 3.2. *The measure d is said to be a distance metric of similarity if and only if:*

- $d(x,y) \geq 0$ *with equality attained* $\iff x = y$
- $d(x,y) = d(x,y)$
- $d(x,y) \leq d(x,z) + d(z,y)$

3.2.1 Hamming Distance

The Hamming distance between two strings or vectors of equal length is the number of positions at which the corresponding symbols differ i.e. the minimum number of substitutions required to change one string into the other.

$$HD(x,y) = \sum_{i} \nVdash(x_i \neq y_i) \tag{1}$$

In texts, the strings are usually not of the same length and thus this metric cannot be used directly. The Hamming distance can be used if the words in a collection of scripts are aligned and turned into individual characters to produce a multiple alignment sequence as in bioinformatics, however one loses much information in this manner. A better way of calculating distance matrices would be to use a distance measure that can work with strings of arbitrary lengths.

3.2.2 (Demerau-)Levenshtein Distance

The Levenshtein distance is a generalisation of the Hamming Distance used for measuring the difference between two sequences of arbitrary lengths. It is the minimum number of insertions, deletions or substitutions required to change one word into the other. This can be further extended to the Demerau-Levenshtein distance by allowing for transpositions for adjacent characters. The Demerau-Levenshtein distance [23] between two strings x and y and corresponding character positions i and j is defined as:

$$LD_{x,y}(i,j) = \begin{cases} 0 & \text{if } i = j = 0, \\ LD_{x,y}(i-1,j)+1 & \text{if } i > 0, \\ LD_{x,y}(i,j-1)-1 & \text{if } j > 0, \\ LD_{x,y}(i-1,j-1)+\mathbb{1}(x_i \neq y_j) & \text{if } i,j > 0, \\ LD_{x,y}(i-2,j-2)+\mathbb{1}(x_i \neq y_j) & \text{if } i,j > 0 \text{ and } x_i = y_{j-1} \text{ and } x_{i-1} = y_j, \end{cases}$$

The first case is the trivial case of empty strings, the second case corresponds to a deletion from x to y, the third case to an insertion from x to y, the fourth case corresponds to a match or mismatch, depending on whether the respective characters are the same whereas the final fifth case (which distinguishes the Demerau-Levenshtein from the regular Levenshtein distance) corresponds to a transposition between two successive characters.

Algorithm 1 Demerau-Levenshtein Distance Algorithm

```
1: procedure DemerauLevenshteinDistance($s, t$)        ▷ $s$ and $t$ are the input strings
2:     $m \leftarrow$ length of string $s$
3:     $n \leftarrow$ length of string $t$
4:     Initialize a 2D array $D$ with dimensions $(m+1) \times (n+1)$
5:     for $i$ from 0 to $m$ do
6:         $D[i][0] \leftarrow i$
7:     end for
8:     for $j$ from 0 to $n$ do
9:         $D[0][j] \leftarrow j$
10:    end for
11:    for $i$ from 1 to $m$ do
12:        for $j$ from 1 to $n$ do
13:            if $s[i] = t[j]$ then
14:                $cost \leftarrow 0$
15:            else
16:                $cost \leftarrow 1$
17:            end if
18:            $D[i][j] \leftarrow \min(D[i-1][j]+1, D[i][j-1]+1, D[i-1][j-1]+cost)$
19:            if $i > 1$ and $j > 1$ and $s[i] = t[j-1]$ and $s[i-1] = t[j]$ then
20:                $D[i][j] \leftarrow \min(D[i][j], D[i-2][j-2]+cost)$        ▷ Transposition
21:            end if
22:        end for
23:    end for
24:    return $D[m][n]$        ▷ Return the Demerau-Levenshtein distance
25: end procedure
```

3.2.3 Jaro(-Winkler) distance

The Jaro similarity [21] is

$$J(x,y) = \begin{cases} 0 & \text{if } m = 0, \\ \frac{1}{3}\left(\frac{m}{|x|} + \frac{m}{|y|} + \frac{m-t}{m}\right) \end{cases}$$

where m is the number of matchings and t the number of transpositions. This score is 0 when the strings do not match at all, and 1 when they match exactly. Each character in one string is compared with all its matching characters in the other and two characters are considered matching only if they are the same and less than $\lfloor\frac{\max(|x|,|y|)}{2}\rfloor - 1$ characters apart. The Jaro distance is thus $1 - J(x, y)$. The Winkler similarity can be extended to the Jaro-Winkler similarity [22] by adding a term to favour strings whose first characters match:

$$JW(x, y) = J(x, y) + lp(1 - J(x, y)) \tag{2}$$

l is the length of common prefix at the start of the string up to a maximum of 4 characters whereas p is a constant scaling factor to adjust the score upwards when having common prefixes and must be at most $\frac{1}{l}$ to ensure the distance remains normalised. The Jaro-Winkler distance is thus $1 - JW(x, y)$.

3.2.4 Jaccard Similarity

Jaccard Similarity is a similarity measure between two vectors of arbitrary lengths and is commonly used to compute the similarity of two items, such as text documents. For two text documents the Jaccard Similarity is:

$$J(\boldsymbol{x}, \boldsymbol{y}) = \frac{|\boldsymbol{x} \cap \boldsymbol{y}|}{|\boldsymbol{x} \cup \boldsymbol{y}|}. \tag{3}$$

The Jaccard Distance can thus be written as:

$$1 - J(\boldsymbol{x}, \boldsymbol{y}) = \frac{|\boldsymbol{x} \cap \boldsymbol{y}| - |\boldsymbol{x} \cup \boldsymbol{y}|}{|\boldsymbol{x} \cup \boldsymbol{y}|}. \tag{4}$$

3.2.5 Cosine Distance

The cosine similarity is a of similarity measure between two non-zero vectors and is defined as the cosine of the angle between the vectors:

$$(cos)(\theta) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{|\boldsymbol{x}||\boldsymbol{y}|} \tag{5}$$

Cosine similarity is a popular metric used to measure the text-similarity between two documents irrespective of their size in Natural language Processing. Words are represented in a vector form and the text documents are represented as a n-dimensional vector space.

3.2.6 Term frequency–inverse document frequency

Term frequency–inverse document frequency (tf-idf) is an efficient numerical statistic which reflects the importance of a word in a document relative to a collection of documents. It is commonly used in natural language processing and information retrieval.

The term frequency $tf(t,d)$ is the relative frequency of term t in document d:

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}, \tag{6}$$

where $f_{t,d}$ is the the number of occurrences of t in document d.

The inverse document frequency measures the informativeness or rarity amongst documents of a term and is defined as:

$$idf(t,D) = log\frac{1+N}{1+|\{d \in D : t \in d\}|}, \tag{7}$$

where N is the number of documents, D is the set of documents. The ones are added to avoid division by zero errors.

Combining these two measures yields the tf-idf:

$$tf - idf(t,d,D) = tf(t,d) \times idf(t,D) \tag{8}$$

The weight of a term increases with the number of occurrences and decreases with the document frequency of the term in the entire collection of documents. Thus, the weights tend to give less importance to common terms.

3.3 Distance Matrix Methods

Once the distances between the different texts are calculated, the distance matrix can then be used to generate the network or tree.

3.3.1 UPGMA (Unweighted Pair Group Method with Arithmetic Mean)

UPGMA is an agglomaration algrothm which computes a rooted tree from an input distance matrix. In the first iteration, it finds the pair of taxa x, y which have the minumum distance, clusters them together as $\{x,y\}$, and the distance from $\{x,y\}$ to every other taxon z is calculated as the average of $d(x,y)$ and $d(x,z)$:

$$d(\{x,y\},z) = \frac{d(x,z)+d(y,z)}{2}. \tag{9}$$

In the subsequent iterations, the elements are clusters that are either singletons (the original taxa) or sets of two or more taxa. In each iteration, the pair of clusters A and B with the smallest distance are clustered together. The distance matrix is then updated by removing the rows and columns for A and B and replacing them with a row for $A \cup B$:

$$d(A \cup B, C) = \frac{|A|d(A,C) + |B|d(B,C)}{|A| + |B|}. \tag{10}$$

This process is repeated until all the taxa are merged into a single cluster which produces a rooted tree. The distance matrix can be updated in a variety of ways such as by simply removing the row and column corresponding to one of the closest two taxa and this has an impact on the output tree. However, the main idea of this algorithm is to iteratively find the closest pair of clusters and replace them with a new cluster.

All agglomerative methods which clusters the closest taxa would fail on any input distance matrix which has its closest taxon not being siblings. Moreover, for all variants of UPGMA, given that it follows such an algorithmic design allows one to determine the first closest pair up to ties, thus if the dataset has only three leaves one can determine the rooted tree that will be constructed and if the dataset has four leaves then one can determine the unrooted tree it will produce. The UPGMA algorithm produces rooted trees and assumes an ultrametricity tree in which the distances from the root to every branch tip are equal, a condition which is almost always violated in stemmatics, thus the root will usually be incorrect.

3.4 Neighbour Joining

Neighbor joining is one of the most widely distance-based method due to its accuracy and even to this day research into why it performs so well in phylogenetics as well as its statistical properties is still ongoing [**?**]. Neighbor joining is also an iterative agglomerative technique wherein the tree is built from the bottom up. The input is an $n \times n$ dissimilarity matrix d and in the first iteration, the n leaves are all in their own clusters. In subsequent iterations, each cluster is a set of leaves however the clusters are disjoint. At the beginning of each iteration the taxa are partitioned into clusters each of which has a rooted tree that is leaf-labeled by the elements in the cluster. During each iteration a pair of clusters is selected to be made siblings resulting in the cluster's respective trees to be merged into a rooted tree by making their roots siblings. Given three sub-trees, the three sub-trees are merged into a tree on all the taxa by adding a new node, r, and making the roots of the three sub-trees adjacent to r. Once the tree is generated the root is ignored such that neighbor joining yields an unrooted tree. The advantage of Neighbor joining over UPGMA is that it chooses which pair of clusters to make siblings (and how to update the distance matrix) using a more sophisticated strategy:

Initialisation: Compute the $n \times n$ Q matrix:

$$Q_{ij} = (n-2)d_{ij} - \sum_{k=1}^{n}(d_{ik} + d_{jk}). \tag{11}$$

Whilst $n > 3$: Find the pair i, j minimising Q_{ij} and call that pair a, b. Make the rooted trees associated with taxa a and b siblings, and call the root of the tree formed u. Update the distance matrix by deleting the rows and columns for a and b, and including a new row and column for u and set $d_{u,k} = \frac{d_{ak}+d_{bk}-dab}{2} \forall k \neq u$. Subtract n by 1.

If $n = 3$; return the star tree with a single internal node v where the roots of the three rooted trees are all adjacent to v.

Algorithm 2 Neighbour Joining Algorithm

```
 1: procedure NeighbourJoining($D$)                     ▷ $D$ is the distance matrix
 2:     $n \leftarrow$ number of taxa in $D$
 3:     Initialize an empty tree $T$
 4:     Initialize a list of taxa $L$ containing all taxa
 5:     while $|L| > 2$ do                               ▷ Until only two taxa remain
 6:         Compute total branch lengths $L_i$ for each taxon $i$
 7:         Compute the $Q$ matrix from distance matrix $D$
 8:         Find the minimum element $q_{ij}$ of $Q$
 9:         Choose taxa $i$ and $j$ such that $q_{ij}$ is minimum
10:         Create a new node $u$ with edge lengths $d_{iu} = \frac{D_{ij}+L_i-L_j}{2}$ and $d_{ju} = D_{ij} - d_{iu}$
11:         Remove taxa $i$ and $j$ from $L$
12:         Attach taxa $i$ and $j$ to node $u$
13:         Update distance matrix $D$ with $u$
14:         Add node $u$ to tree $T$
15:     end while
16:     Attach the two remaining taxa to $T$ with the appropriate edge lengths
17:     return $T$                                       ▷ Return the phylogenetic tree
18: end procedure
```

Neighbour Joining is very efficient compared to other methods which makes it useful when dealing with large data sets as well as for bootstrapping. Unlike UPGMA it is statistically consistent under many models of evolution meaning that given enough data, it reconstructs correct phylogenetic trees with high probability [**?**], however since it cannot generate general networks, it does not faithfully reconstruct stemmas especially in the presence of contamination. Another advantage neighbor joining has over UPGMA is that it does not assume ultrametricity which also makes it far more compatible in a stemmatological context. Its main downside is that it lacks tree search and optimality criteria meaning there is no guarantee that the recovered tree is optimal. Thus usually Neighbour Joining is used to produce a tree which is then used as an initial guess in a tree search using some optimality criterion [**?**].

3.5 Parsimony Based Methods

Maximum parsimony is another concept which can be used to construct trees. The output is a tree T in which the input sequences are placed at the leaves of T and additional sequences are placed at the internal nodes of T such that the total number of changes over the entire tree, is minimised. Another way of defining maximum parsimony is as the Hamming Distance Steiner Tree Problem wherein the input is a set of sequences and the output is a tree connecting these sequences at the leaves with other sequences (the Steiner points) at the internal nodes, which minimises the total of the Hamming distances on the edges of the tree. The Hamming distance between two sequences of the same length is the number of positions in which they disagree, thus the sum of the Hamming distances on the edges of the tree is equal to its tree length. Determining the optimal tree under this criterion is an NP-hard problem [**?**], and hence heuristics are used to find suitable solutions which are not necessarily globally optimal. Maximum parsimony heuristics are computationally very intensive on large datasets. Moreover, this method has been proven to be statistically inconsistent under standard DNA sequence evolution models, and may even converge to the wrong tree as the sequence length increases.

Given a tree T and character state assignments to the leaves of T. The aim is to find character state assignments to the internal nodes of T so as to minimise the number of edges with different states at the edge's endpoints. This is the "fixed tree parsimony problem" or the "small parsimony problem" to distinguish it from the "large parsimony problem," which is to find the optimal tree and its internal node labels achieving the optimal parsimony score. Since only the total number of changes is counted, the parsimony score of a tree is not affected by the tree's rootedness. The input is an unrooted binary tree T with set of leaves S, and character $c : S \to \{1, 2, ..., r\}$ whilst the output is the assignment of character states to the internal nodes of T such that the number of edges $e = (u, v)$ where $c(u) \neq c(v)$ is a minimum. For binary trees as in phylogenetics the Fitch Margoliash algorithm can be used to calculate the parsimony score and determine an optimal labeling of the internal nodes. For arbitrary trees the Hartigan algorithm is used.

3.6 Minimum Evolution

Minimum evolution is a collection of methods. The input is an $n \times n$ dissimilarity matrix d, and every tree topology on n leaves is assigned edge weights using some version of least-squares. The total length of each tree is then computed by adding up the lengths of all the branches in the tree. The tree T with its edge weighting w which minimises $\sum_{e \in E(T)} w(e)$ is returned. Such methods are distinguished by how they define the edge weighting of each tree. For example, given a tree T and an input dissimilarity matrix d, the optimal weights on the edges could be based on minimizing the L_2 (OLS) or the weighted L_2 (WLS) distances. Since least squares is used, this runs in polynomial time. On the other hand, finding the tree with the minimum total branch length is generally much harder. For example, it is NP-hard under the OLS criterion for integer branch lengths and under a variant of the WLS criterion called "balanced minimum evolution".

3.7 Minimum Spanning Trees

In [12], instead of phylogenetic tree building algorithms, minimum spanning trees are constructed using Kruskal's algorithm. A minimum spanning tree is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices without any cycles and with the minimum possible total edge weight. A feature of this approach is that it does not attempt accounting for missing taxa, resulting in a tree consisting only of the observed variants.

Algorithm 3 Kruskal's Algorithm for Minimum Spanning Tree

```
1: procedure KRUSKAL($G$)                              ▷ $G$ is the graph
2:     $T \leftarrow \emptyset$                        ▷ Initialize an empty set for the MST
3:     Sort all edges of $G$ in non-decreasing order of their weights
4:     for each vertex $v$ in $G$ do
5:         MakeSet($v$)
6:     end for
7:     for each edge $(u, v)$ in $G$ in sorted order do
8:         if FIND($u$) $\neq$ FIND($v$) then           ▷ If adding $(u, v)$ doesn't create a cycle
9:             $T \leftarrow T \cup \{(u, v)\}$        ▷ Add edge $(u, v)$ to MST
10:            UNION($u, v$)                           ▷ Merge the sets containing $u$ and $v$
11:        end if
12:    end for
13:    return $T$                                      ▷ Return the MST
14: end procedure
```

4 Statistical Methods

Distance based methods are not considered state of the art in phylogenetics anymore but are instead used to generate an initial guess tree which is then used to infer the tree using statistical methods.

4.1 Maximum Likelihood Based Methods

Maximum likelihood Estimation is a very popular statistical technique where an explicit model of nucleotide substitution is used to evaluate trees. The model tree consists of a tree topology T and a set of parameters θ. MLE takes as input a set S of sequences of identical lengths, and finds the model tree $(\hat{T}, \hat{\theta})$ which maximises the probability that the sequences occur given the input tree T and the parameters θ $\mathbb{P}[S|T^*, \theta^*]$. The only difference between MLE approaches with different evolution models are the parameters. Since MLE requires the estimation of all the model parameters, increasing the number of parameters rapidly increases the computational demands and the risk of over-fitting. Moreover, whilst the MLE the differences in scores between different trees can be insignificant and thus calculating the optimal and the near-optimal trees is sometimes the objective. Moreover, MLE algorithms are not always statistically consistent. Finding an optimal ML tree is an NP-hard problem, thus requiring the use of heuristics rather than finding the exact solution. Since the number of trees on n grows exponentially with n, examining each possible tree topology is not feasible and thus heuristic searches are used to explore the treespace. Heuristic searches for ML are more complicated than heuristic searches for maximum parsimony because even scoring a tree is computationally intensive as the estimation of numeric parameters on a tree is also computed heuristically and does not guarantee global optima. Hence, ML phylogeny estimation is complicated and computationally intensive, even for basic sequence evolution models. The evolution of text is far more complicated than the evolution of genetic sequences due to the larger number of possible characters, various different mechanisms by which errors occur. Unlike genetic changes which are usually assumed to be independent, in textual evolution spatial correlations are very common. Moreover, the fact that the sequences are typically much longer, the multifurcating nature of textual evolution and the fact that not all extant scripts could be placed in the leaf nodes of the tree vastly increase the computational demands. Evolution models are defined by their parametric transition probability matrices, which in the case of DNA sequences can be expressed in the form

$$P(t) = \begin{pmatrix} P_{A\to A}(t) & P_{G\to A}(t) & P_{C\to A}(t) & P_{T\to A}(t) \\ P_{A\to G}(t) & P_{G\to G}(t) & P_{C\to G}(t) & P_{T\to G}(t) \\ P_{A\to C}(t) & P_{G\to C}(t) & P_{C\to C}(t) & P_{T\to C}(t) \\ P_{A\to T}(t) & P_{G\to T}(t) & P_{C\to T}(t) & P_{T\to T}(t) \end{pmatrix} \tag{12}$$

where $P_{A\to A}(t)$ is the probability that a site in state x evolves into state y in time t. There are various ways to define the transition probabilities and similar models exist for protein sequences however, although textual evolution can be modeled in a similar manner, such models for textual evolution are not well established. In textual evolution time does not have the same role as in biological evolution since the existing manuscripts can be used as sources for copying after an arbitrarily long time and thus copies from older manuscripts cannot be assumed to contain more errors than a copies made of newer manuscripts.

4.1.1 Bayesian Methods

Bayesian methods are similar to ML methods in that they also calculate likelihoods of trees based upon explicit parametric mathematical models of evolution. Instead of attempting to find the model tree with the largest probability of generating the observed data, Bayesian methods sample from the set of model trees with frequency proportional to their likelihoods, given the observed data. The result is that Bayesian methods produce a set of trees rather than a single tree which is then used to evaluate the statistical support for different evolutionary hypotheses. Bayesian methods are usually implemented using Markov Chain Monte Carlo (MCMC) techniques where a Bayesian method performs a random walk through the space of model trees. Each time it visits a model tree, it computes the probability of the observed data, using Felsenstein's Pruning Algorithm. The probability of accepting the new model tree depends on whether the probability has increased or decreased; for example, in Metropolis–Hastings MCMC, the new tree may be accepted even if the probability has decreased. Whenever the new tree is accepted, the MCMC chain continues from the new model tree. The Bayesian MCMC walk is guaranteed to converge to the stationary distribution on model trees, as long as the MCMC walk satisfies the detailed balance property, which essentially says that $\pi(i)P_{ij} = \pi(j)P_{ji}$, where $\pi(x)$ is the equilibrium probability of state x and P_{xy} is the probability of moving to state y from state x. After the MCMC chain has run long enough to become stationary, a sample of the model trees it visits is then taken from the trees visited after some "burn-in" period. If a point estimate of the tree topology is desired, then a summary of the distribution such as a consensus tree can be computed using various techniques. The frequency with which a tree topology appears in the sample is called the posterior probability. Another common summary statistic is the maximum a posteriori (MAP) tree, which is the tree topology that appears the most frequently in the sample. A third approach assembles a set of tree topologies, starting with the most frequently observed trees and then decreasing, until a desired threshold is achieved (e.g., where 95 percent of the probability distribution is in the set), and then computes a point estimate based on this subset of trees. Each point estimate can then be used as an estimated phylogeny, and the support for the branches of the phylogeny can similarly be computed based on the observed distribution.

There are several challenges in using Bayesian methods. Bayesian methods require priors, and the choice of priors can affect the accuracy of the resultant trees. Another disadvantage is that Bayesian MCMC methods need to be run until they have reached the stationary distribution, and it is not at all straightforward to assess whether stationarity has been reached. On larger datasets Bayesian methods become computationally expensive as it typically increases the time to reach stationarity.

4.1.2 Markov-Chain Monte Carlo

5 Reliability

Once a tree is generated, it cannot be assumed to be reliable. Typically, scholars use various different methods constructing the tree and place more confidence in subgraphs that are consistently recovered with different methods. A consensus tree is a summary or representation of the phylogenetic relationships derived from multiple individual phylogenetic trees.

In phylogeneticists often generate multiple trees to explore the uncertainty and variability in evolutionary relationships among a group of organisms. These individual trees can result from different methods, datasets, or variations in the analysis parameters.

The process of constructing a consensus tree involves combining information from multiple trees to produce a single, summarized representation that reflects the common or well-supported aspects of the phylogenetic relationships.

5.1 Bootstrapping

Bootstrapping is often used to estimate the statistical reliability of a tree's subgraphs by taking the original dataset and picking the same number of sites randomly from within the dataset with replacement, some sites are sampled once or more than once whereas others are not sampled at all. This is repeated using the original dataset to generate a large number of different auxiliary datasets. The tree inference method is then applied to each of the auxiliary datasets and the corresponding output trees are compared to the tree from the original dataset. The percentage of subgraphs which occur in the auxiliary trees and in the original tree are then calculated and assigned to the nodes which define the subgraphs in the original tree. A high boostrap value indicates a high level of confidence that a particular subgraph is independent of the sites used to calculate the tree. Applying the tree inference method to each auxiliary dataset takes the same amount of time as calculating the original tree, hence there are computational constraints for large datasets or computationally demanding models. One advantage of Bayesian methods is that they automatically generate confidence values for the subgraphs in the tree, obviating bootstrapping.

6 Rooting A Stemma

There are two main methods of rooting a phylogenetic tree: using an outgroup or a molecular clock. In the outgroup method one includes a taxon that is close enough to the set of taxa under investigation (ingroup) to make comparisons however farther apart from all ingroup taxa are to eachother. An alternative is to use a ”molecular clock” where it is assumed that the same amount of changes are observed on all lineages implying that there should be a point in the tree with equal branch lengths from that point to all leaves. The root is the node which makes the amounts of change approximately equal on all lineages. In stemmatology, the outgroup is rarely available and a molecular clock assumption would almost always be violated since scripts evolve much more dynamically than genetics.

6.1 Minimum Cost Heuristic

In [12], a rooting method which may be applicable to stemmatology is suggested. Given a non-oriented tree, one can assign any of the nodes as the root and calculate the tree cost, that being the sum of the edge weights of the potential root to the other nodes in the tree, with all nodes being considered once as a potential root. The root is then taken as the node with corresponding with the minimal cost tree. If two or more nodes generate trees of minimal cost, one of them is randomly selected as the root of the reconstructed tree.

A drawback of this method is that it assumes that textual evolution leads to trees which tend to balance out over time with respect to the differences between the nodes. When the stemma of a given document does not branch, this method tends to fail as a tree with no branchings will have higher costs than one with branchings. Another drawback is that this method assumes that the stemma had only one root which may not be the case in stemmatology.

6.2 Edit Directionality

[**?**] suggests a different method which relies on the directionality of changes. Correctly guessing all directions of the edges in a tree would naturally lead to the root. Some changes are less likely to occur than others. Adding text is less likely to occur than transpositions and deletion of text and thus, from this one can guess the direction of textual evolution. However due to randomness, it is not recommendable to use the directions between each pair of nodes as then one would end up with an unrealistic number of roots and usually a non-DAG structure. Thus instead of using single edges or the entire graph, [**?**] suggests using the paths from every leaf to every other leaf. Starting from an unrooted tree. For any leaf, suppose that there is at least one other leaf which remote enough that their latest common ancestor is the root. It is assumed that any shortest path from leaf to leaf in the unrooted tree passes through the latest common ancestor of both leaves which is denoted L. All edges towards the two leaves should point away from L and hence traversing the path by edges from one leaf to the other, the direction of changes in the rooted tree changes when passing L. Assuming that for each leaf there exists at least one other leaf for which L is the root, then, comparing all paths starting from a leaf i, the L most distant from i must coincide with the root since there can be no later common ancestor in the rooted tree. Furthermore, all leaves should converge on the same node as the most distant L, hence should one detect that node on any path which changes directionality, one would know for each such path the direction change point L and thus the root.

7 Accuracy Measures

The methods are evaluated based on their success of finding a stemma that is close to the true stemma. We use the Average Signed Distance suggested in [6] and some of the measures used in multimedia phylogeny suggested in [13] [12].

7.1 Leaves

This metric evaluates whether the leaves of the reconstructed stemma are the same as in the true stemma.

$$Leaves(T_1, T_2) = \frac{Leaves(T_1) \cap Leaves(T_2)}{Leaves(T_1) \cup Leaves(T_2)} \tag{13}$$

7.2 Depth

The depth distance is defined as the number of edges between the true root of the true stemma and the predicted root in the reconstructed tree. Defining $dist(i, j, T)$ to be the number of edges between nodes i and j on the tree T, the Depth is then:

$$Depth(T_1, T_2) = dist(Root(T_1), Root(T_2), T_2). \tag{14}$$

7.3 Indirected Edges

This measure evaluates the edges. Thus, should an edge connect two nodes in the reconstructed tree and in the the original, then it is considered correct:

$$Edges(T_1, T_2) = \frac{Edges(T_1) \cap Edges(T_2)}{n - 1} \tag{15}$$

7.4 Directed Edges

This measure evaluates the edges along with their direction. Thus, should a directed edge connect two nodes in the reconstructed tree and in the the original, then it is considered correct:

$$DirectedEdges(T_1, T_2) = \frac{DirectedEdges(T_1) \cap DirectedEdges(T_2)}{n - 1} \tag{16}$$

7.5 Ancestry

This measure evaluates whether the ancestors (parents, grandparents etc.) of each root are the same in the reconstructed tree and in the true tree.

$$Ancestry(T_1, T_2) = \frac{Ancestry(T_1) \cap Ancestry(T_2)}{Ancestry(T_1) \cup Ancestry(T_2)} \tag{17}$$

7.6 Average Signed Distance

In situations where there are missing manuscripts or when one uses methods which include internal nodes, two arbitrary latent tree structures are being compared and thus the usual measures such as counting the number of shared edges do not apply since there is no one-to-one correspondence between the latent nodes in the true stemma and the estimated one. Moreover, the tree generating algorithms which place the extant taxa as leaves makes the 'Leaves' measure redundant. It is not guaranteed that the number of latent nodes will be the same, let alone their positions in the stemma. Thus, we employ the average sign similarity score [15].

This measure depends on the number of edges between pairs of nodes and ignores possible edge length information. For each pair of nodes A, B, the true distance, $d(A, B)$ is defined as the number of edges on the shortest path between A and B in the true stemma. Thus $d(A, B) = d(B, A)$. Similarly, the same quantity computed from a hyptothetical stemma is denoted by $d'(A, B)$. Whenever the hypothesised stemma is correct, $d'(A, B) = d(A, B)$ for all pairs of nodes, and otherwise the two values differ for some A and B. Given three nodes A, B, and C, we can measure the distances $d(A, B)$ and $d(A, C)$. We consider which one of these two distances is greater than the other, or whether they are equal. Let $sign(d(A, B) - d(A, C))$ be the sign of the difference between the two distances, so that the index:

$$u(A, B, C) = 1 - \frac{1}{2}|\text{sign}(d(A, B) - d(A, C)) - sign((d'(A, B) - d'(A, C))|, \tag{18}$$

so that:

$$u(A, B, C) = \begin{cases} 1 & \text{if } sign(d(A, B) - d(A, C)) = sign(d'(A, B) - d'(A, C)) \\ 0 & \text{if } sign(d(A, B) - d(A, C)) = -sign(d'(A, B) - d'(A, C)) \\ 1/2 & \text{otherwise.} \end{cases}$$

The average sign similarity score is the average of over all distinct observed nodes corresponding to the extant manuscripts (but not the latent nodes) given that none of the three nodes are equal to each other. This measure can be used with any pair of graphs, given that both include all the observed nodes and may include any number of additional nodes, which need not be identical for the correct and the proposed stemma. The computation of the distance requires that the pairwise distances between all nodes are computed and recorded in a table. After this, the average can be computed.

8 Missing Data

In phylogenetic data one often deals with missing data. Typically, these values are replaced by character states to obtain the best possible score. An alternative treatment of missing data replaces all the missing entries with the same new state, and then seeks a tree that optimizes the criterion. If a dataset containst too many missing characters, the corresponding poistions are removed before the tree is computed. In stemmatology missing data could be due to either destruction of part of a manuscript or due the scribe skipping certain words or punctuation, in which case the changes are informative. In the case that large sections of a manuscript is missing it is usually best to either remove the variant, or to only use the available part. Another option would be to impute the missing sections from the closest manuscripts.

9 Method

9.1 Phylogenetics and Stemmatology Tools

9.1.1 CollateX

CollateX [25] is a software tool designed for collating, aligning and visualizing multiple versions of a text, especially for tasks related to textual criticism and historical text analysis. It is often used to compare different manuscript versions, editions, or translations of a text and identify variations or differences between them.

The input texts are assumed to have been tokenized. During alignment, CollateX matches tokens across different text versions, aligning identical tokens and inserting placeholders for unmatched ones to ensure that the sequences line up. The alignment might also involve identifying and handling transpositions, where parts of the text have been moved.

9.1.2 PAUP*

PAUP* is one of the most widely used phylogenetic program applied to stemmatology. It offers an easy-to-use graphical interface with many options, algorithms, and parameterisation options. Methods include maximum Likelihood, parsimony based methods, distance matrices as well as bootstrap and consensus trees.

9.1.3 Mr Bayes

MrBayes is a Bayesian phylogenetics program and is considered the standard model of choice across a wide range of phylogenetic and evolutionary models. MrBayes relies on variants of Markov chain Monte Carlo (MCMC) methods to estimate the posterior distribution of model parameters.

9.1.4 SplitsTree

SplitsTree is a popular program for computing unrooted phylogenetic networks from molecular sequence data. As an input it takes distance matrices or a set of trees and outputs a phylogenetic tree or network. Methods include split decomposition, neighbor-net, consensus networks, super networks methods or methods for computing hybridization or simple recombination networks. This makes it able to detect contamination however the graph outputs are generally very difficult to interpret.

9.2 Data Sets

To test the methods, two stemmatology datasets which were previously used in a computational stemmatology challenge [24] (Parzival and Heinrichi) were used along with some artificially created datasets. [24] also includes two other traditions, however the true stemmas are not known. On top of this Phareta Fidei was studied although the true stemma is unknown.

9.2.1 Real Data Sets

- Parzival: This dataset is comprised of 16 documents consisting of the first part of Wolfram von Eschenbach's German poem Parzival, translated to English by A.T. Hatto, and copied by hand by scribes.

- Heinrichi: This data set is comprisd of 67 variants of an old Finnish text (Piispa Henrikin surmavirsi 'The Death-Psalm of Bishop Henry'). It was artificially constructed by volunteer scribes, who copied a given text by hand, according to an imaginary stemma. To simulate the situation where a portion of the manuscritps are missing 30 of the variants were left out of the dataset and some of the manuscripts also had some significant passages deleted to simulate the situations where manuscripts are partially lost.

- Phareta Fidei: This dataset is of a real Medieval anti-Semitic tradition written in Latin, composed of 10 variants obtained from different libraries and archives. On average, the texts have about 225 tokens about a quarter of which are non-constant. The texts vary in their spelling, abbreviations, sentence structure as well as word count.

9.2.2 Artificial Datasets

A synthetic datasets consisting of near-duplicate documents were created according to a variety of of parameters. Subsets of the Reuters_50_50 [?] training dataset were used depending on the word-count of the article. Artificial stemmas were generated by taking a document to be the root and generating a set of child documents with randomised edits. Each child document is modified up to an editing limit. The edit operations were selected such that the overall meaning of the child documents does not stray far away from the original. These are:

- Synonym exchange: A given word is replaced by a randomly selected synonym using the WordNet [?] dictionary.
- Misspelling and Correction: A word is misspelled or a misspelling is corrected using the Birkbeck [30], Holbrook [31], Aspell [29] and Wikipedia [28] misspelling datasets obtained from [30]
- Insertion and removal of adjectives and adverbs: Adverbs and adjectives are either added or removed before nouns. A subset of the Corpus of Contemporary American English (COCA) [33] n-gram dataset was used to add the adverbs and adjectives whilst the NLTK [?] package was used to remove them by checking their part-of-speech tags.
- Insertion and removal of sentences: Initially, some sentences of the original document are held out before the tree generation. A random sentence from the held held-out set and returning it to the document at the same position as it would have occurred originally to preserve the meaning. Removal of sentences is also random. Once a sentence is removed, it does not occur in any of the descendants.
- Word/sentence doubling and deletion of doubled words/sentences. A random word/sentence is doubled. Doubled words/sentences are deleted with a high probability in the descendent node.

- Contamination. A document is created using the first few paragraphs from one document and the rest from another. Other edits can be added afterwards.

9.3 Phylogenetics Software Pipeline

The texts were all in ".txt" format and processed through the python package CollateX which generates the best alignment of the tokens in the text. As part of the preprocessing for using the ready-made phylogenetics packages (MrBayes, PAUP* and SplitsTree) the texts were then reformatted into the Nexus format by ordering the words at each position in the text in alphabetical order. Each variant of the token in each position was then assigned symbol $(A, B, C, ...)$, such that if say, three different variants of a given word appear in the texts, then the symbols A,B, and C appear in the Nexus file at the same position, identical words at the same position are assigned the same symbol, whereas missing tokens are represented by a question mark. Modern philologists typically keep punctuation (hyphens, commas, periods etc.) since in some cases they may be informative. Having said that, punctuation is typically heavily dependent on the scribes' writing style and thus some philologists remove punctuation since its more random nature can be seen as adding noise to the data [1]. In our analysis we keep the punctuation as part of words since if one considers punctuation as separate tokens, it would be much more dominant over words even though it is not as informative.

Figure 3 is an example of a Nexus datablock. "ntax" corresponds to the number of taxons, "nchar" corresponds to the number of characters in the aligned sequences which must be the same for all taxons (in our case the symbols correspond to the word variants and "?"'s correspond to gaps in the alignment or missing words) and "format symbols" is the list of symbols used to denote the words.

```
#NEXUS
BEGIN DATA;
    dimensions ntax = 5 nchar = 10;
    format symbols = "ABCD" labels = left;
    matrix
        Taxon1  AABAAAAAAB
        Taxon2  AABBA?BABA
        Taxon3  AAABAACACA
        Taxon4  AAA?AAABCA
        Taxon5  AAABAACBDA;
end;
```

Figure 3: Example Nexus File Data Block.

The nexus files are then fed into phylogenetics software such as PAUP* and the Newick tree outputs are then read using the Bio.Phylo package [26] in python, converted to Networkx [27] objects and finally compared to the true stemma via the averaged signed distance since the other measures of accuracy are not very ideal in this case due to the fact that phylogenetic tree outputs contain latent nodes.

9.4 NLP Distance Matrix Pipeline

To generate the stemmata we use a pipeline similar to that suggested in [12] as shown in Figure 4.

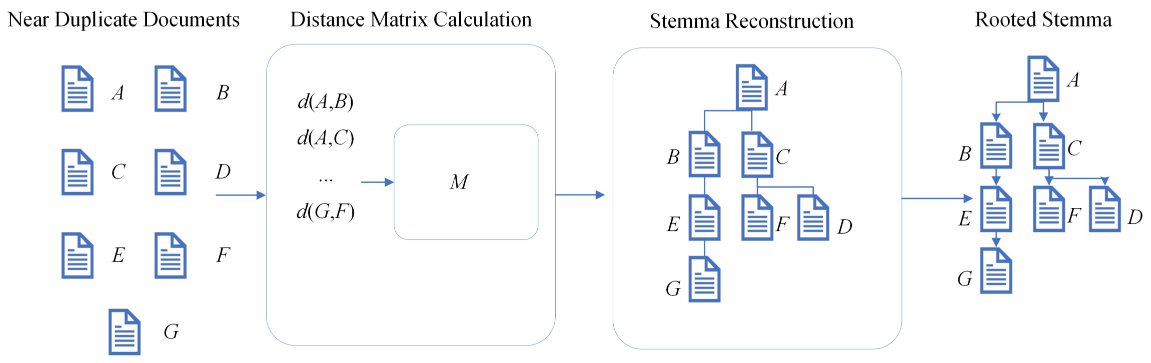


Figure 4: Pipeline for Distance Matrix Calculation. Given N near-duplicate documents, an $N \times N$ symmetric dissimilarity matrix M is calculated and used to construct an undirected tree or graph using a minimal spanning tree or agglomerative algorithm. Finally, heuristics are used to root the tree.

10 Results

11 Discussion

12 Conclusion

12.1 Future Works

References

[1] Roelli, Philipp. (2020). Handbook of stemmatology : history, methodology, digital approaches. Berlin, Germany: De Gruyter.

[2] M. Spencer and C. J. Howe, ”Estimating distances between manuscripts based on copying errors,” Literary and Linguistic Computing, vol. 16, p. 467–484, 2001.

[3] W. M. Fitch, ”Toward defining the course of evolution: minimum change for a specific tree topology,” Systematic Biology, vol. 20, p. 406–416, 1971.

[4] Q. Nguyen and T. Roos, ”Likelihood-based inference of phylogenetic networks from sequence data by phylodag,” in International Conference on Algorithms for Computational Biology, 2015.

[5] A. Lee, ‘Numerical Taxonomy Revisited: John Griffith, Cladistic Analysis and St. Augustine’s Quaestiones in Heptateuchum.’ Studia Patristica XX: 24-32, 1989.

[6] T. Roos, T. Heikkilä, Evaluating methods for computer-assisted stemmatology using artificial benchmark data sets, Literary and Linguistic Computing, Volume 24, Issue 4, December 2009, Pages 417–433, https://doi.org/10.1093/llc/fqp002.

[7] D. L. Swofford, PAUP*. Phylogenetic Analysis Using Parsimony (*and Other Methods). Version 4. Sinauer Associates, Sunderland, Massachusetts. 2003.

[8] J. Felsenstein, PHYLIP (Phylogeny Inference Package) version 3.6. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle, 2005.

[9] F. Ronquist, and J. P. Huelsenbeck. MRBAYES 3: Bayesian phylogenetic inference under mixed models. Bioinformatics 19:1572-1574. 2003.

[10] C. Liu, C. Chen, J. Han, and P. S. Yu, “GPLAG: Detection of software plagiarism by program dependence graph analysis,” Proc. KDD 2006, pp. 872–881, 2006

[11] J. Sun, S. Papadimitriou, C.-Y. Lin, N. Cao, S. Liu, W. Qian, "MultiVis: Content-based social network exploration through multi-way visual analysis," Proc. SDM 2009, pp. 1063–1074, 2009

[12] Marmerola GD, Oikawa MA, Dias Z, Goldenstein S, Rocha A. On the Reconstruction of Text Phylogeny Trees: Evaluation and Analysis of Textual Relationships. PLoS One. 2016 Dec 19;11(12):e0167822. doi: 10.1371/journal.pone.0167822. PMID: 27992446; PMCID: PMC5167249.

[13] Dias Z, Rocha A, Goldenstein S. Image Phylogeny by Minimal Spanning Trees. IEEE Transactions on Information Forensics and Security. 2012; 7(2):774±788. doi: 10.1109/TIFS.2011.2169959

[14] S. Wehner, "Analyzing worms and network traffic using compression," J. Comp. Secur., vol. 15, pp. 303–320, 2007

[15] T. Roos, T. Heikkilä, Evaluating methods for computer-assisted stemmatology using artificial benchmark data sets, Literary and Linguistic Computing, Volume 24, Issue 4, December 2009, Pages 417–433, https://doi.org/10.1093/llc/fqp002

[16] T. Roos, and Y. Zou. 2011. "Analysis of Textual Variation by Latent Tree Structures." In Proceedings of the 11th International Conference on Data Mining (ICDM-2011), edited by Diane J. Cook, 567–576. Los Alamitos, California: IEEE Computer Society. Available at: http://www.cs.helsinki.fi/u/ttonteri/pub/icdm2011.pdf. Accessed 28 October 2015.

[17] N. Friedman 1998. "The Bayesian Structural EM Algorithm." In Uncertainty in Artificial Intelligence: Proceedings of the Fourteenth Conference (1998), edited by Gregory F. Cooper, 129–138. San Francisco: Morgan Kaufmann. Available at: www.cs.huji.ac.il/ nir/Papers/Fr2.pdf Accessed 28 October 2015.

[18] T. Roos, T. Heikkilä and P. Myllymäki. 2006. "A Compression-Based Method for Stemmatic Analysis." In ECAI 2006: Proceedings of the 17th European Conference on Artificial Intelligence: August 29 – September 1, 2006, edited by Gerhard Brewka et al., 805–806. Amsterdam: IOS Press.

[19] Stemmaweb ceators. https://stemmaweb.net/

[20] Damerau, Fred J. (March 1964), "A technique for computer detection and correction of spelling errors", Communications of the ACM, 7 (3): 171–176, doi:10.1145/363958.363994, S2CID 7713345

[21] Jaro, Matthew A. (1 June 1989). "Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida". Journal of the American Statistical Association. pp. 414–420. doi:10.1080/01621459.1989.10478785.

[22] Winkler, William E. (1990). "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage".

[23] Fred J. Damerau (1964). A technique for computer detection and correction of spelling errors. Communications of the ACM, 7: 171–176.

[24] https://www.cs.helsinki.fi/u/ttonteri/casc/data.html

[25] Dekker, R. H. and Middell, G., (2011). Computer-Supported Collation with CollateX: Managing Textual Variance in an Environment with Varying Requirements. Supporting Digital Humanities 2011. University of Copenhagen, Denmark. 17-18 November 2011.

[26] Talevich, E., Invergo, B.M., Cock, P.J. et al. Bio.Phylo: A unified toolkit for processing, analyzing and visualizing phylogenetic trees in Biopython. BMC Bioinformatics 13, 209 (2012). https://doi.org/10.1186/1471-2105-13-209

[27] Aric A. Hagberg, Daniel A. Schult, Pieter J. Swart, Exploring Network Structure, Dynamics, and Function using NetworkX, Proceedings of the 7th Python in Science conference (SciPy 2008), G. Varoquaux, T. Vaught, J. Millman (Eds.), pp. 11–15.

[28] Wikipedia: The Free Encyclopedia. Common Misspellings. Available: `https://en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings/For_machines`

[29] K. Atkinson: GNU Aspell spellchecker. Available: `http://aspell.net/`

[30] Mitton R. Corpora of misspellings for download; 1985. Available: http://www.dcs.bbk.ac.uk/ roger/ corpora.html

[31] Holbrook D. 1964, English for the Rejected: Training Literacy in the Lower Streams of the Secondary School, Cambridge University Press.

[32] Bird S, Klein E, Loper E. Natural Language Processing with Python. O'Reilly; 2009.

[33] Davies M. N-grams data: Corpus of Contemporary American English; 2008. Available: `http://www.ngrams.info`