Implementation of a Phylogenetic Pipeline for Protein Sequence Analysis

Jakub Duda, Soulaimane Salehddine, Mariana Lourenço

May 14, 2024

Introduction

This report outlines the development and execution of a comprehensive phylogenetic analysis pipeline for protein sequences. The pipeline was designed to automate the retrieval of protein sequences, performing of BLAST analysis, conducting multiple sequence alignment (MSA), and ultimately the construction of phylogenetic trees.

Methods

Every subsequent step of the pipeline is implemented as a function and it can either take its input in a form of a parameter or alternatively if no parameter is supplied the input is taken from a file with a predefined name.

Sequence retrieval

The pipeline initiates with the retrieval of the protein sequence for the given UniProt ID supplied to the program by the user. This is facilitated by the Uniprot API, which is used to retrieve data in FASTA format. The obtained sequence is written to a file named `sequence.fasta`.

BLAST Analysis

The retrieved sequence undergoes a BLAST search against the NCBI's non-redundant protein database, filtering out human and synthetic constructs to focus on relevant species. The top 10 hits representing distinct identified species are saved into a file called `sequences_to_analyse.fasta`.

Multiple Sequence Alignment (MSA)

The selected sequences are aligned using the API of the Clustal Omega service, provided by EMBL-EBI. The alignments are obtained in nexus format and are subsequently saved in a file called `alignment.txt`.

Phylogenetic Tree Construction

The phylogenetic tree is constructed from the MSA data using methods such as UPGMA, Neighbour-Joining or Maximum Parsimony, provided by the BioPython toolkit. The tree is visualized and saved in various formats for analysis.

Tools and Packages

- **BioPython:** Utilized for sequence handling, BLAST operations, alignments, and tree construction.
- **UniProt API:** Employed for fetching protein sequences.
- **Clustal Omega via EMBL-EBI:** Used for online MSA.
- **NCBIWWW:** Used within BioPython for online BLAST searches.
- **matplotlib:** Python library used for visualization of phylogenetic trees.
- **requests:** Python library used for interaction with aformentioned APIs.

Results

The phylogenetic analysis pipeline successfully identified and aligned sequences from ten different species and constructed a detailed phylogenetic tree illustrating the evolutionary relationships among these species. The tree analysis highlighted some expected evolutionary patterns as well as novel relationships that may warrant further biological investigation.

Difficulties

Several challenges were encountered during the development and execution of the phylogenetic analysis pipeline:

1. **High Runtime of BLAST Analysis:** The BLAST analysis proved to be time-consuming, severely increasing the total runtime of the pipeline.
2. **Uniqueness of Species in BLAST Results:** Ensuring species uniqueness in BLAST results was complex, necessitating additional parsing logic.
3. **Integration of Tools and APIs:** Compatibility and stability issues arose when integrating various tools and APIs into a single workflow.
4. **Visualization of Phylogenetic Trees:** Some design changes were needed to clearly represent complex evolutionary relationships.

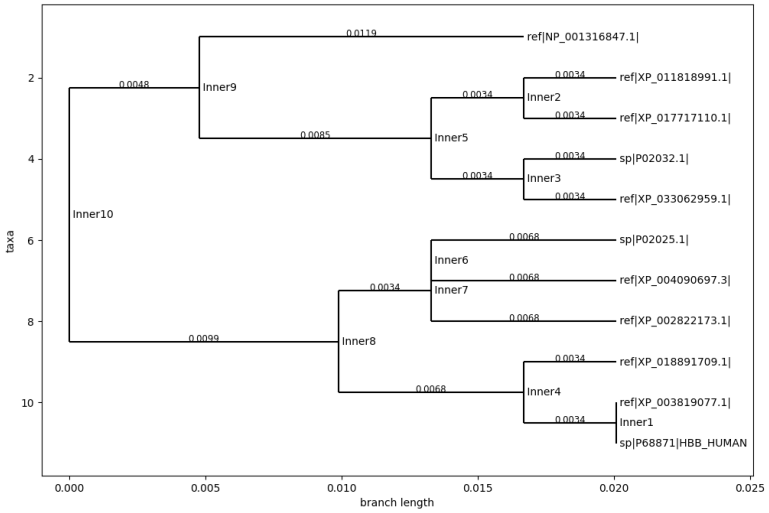

Figure 1: Phylogenetic tree constructed from multiple sequence alignment of protein sequences. The tree uses branch lengths to represent phylogenetic distances, illustrating the evolutionary relationships among the selected protein sequences. Notably, sequences like `sp|P02032.1|` and `ref|XP_033062959.1|` are closely related, suggesting a lesser evolutionary divergence compared to others.

Conclusion

The developed phylogenetic pipeline effectively automates the analysis of protein sequences, from data retrieval to phylogenetic tree construction. This automation supports more efficient and reproducible analyses, contributing valuable insights into biological research and evolutionary studies. The project demonstrates the importance of computational efficiency and robust data handling in bioinformatics, providing a foundational tool for further scientific exploration.