Cross-Species BLASTp Analysis

Automating Protein Sequence Comparisons Across Species

Thornton Alberto-John
Kamikouchi Lab, Fall 2024

Contents

Abstract

Cross-species protein sequence analysis is vital for comparative genomics and evolutionary studies. For example, we needed to compare the amino acid sequence of 48 proteins associated with hearing in 7 drosophila species [3]. However, manually using the BLASTp software from the NCBI website would be very time-consuming. This is also the case when looking at large datasets across several species. To address this, I developed an automated pipeline that integrates BLASTp with Python, automating querying and downloading protein sequences across species. In the following report, I will be solely discussing the development and usage of this pipeline over the actual results of the comparison between the proteins associated with hearing and will merely use that as an example for the use of this pipeline.

This tool accepts gene-specific datasets in Excel format, downloads corresponding protein sequences from NCBI, creates BLAST-compatible databases, and performs cross-species comparisons. The results are generated in CSV format. Also, it downloads FASTA files that can be used with MEGA to build phylogenetic trees.

By reducing manual intervention, this code provides a scalable solution for researchers. Future enhancements could include downloading protein sequences based on gene names instead of specific IDs and further optimising runtime performance.

1 Introduction

1.1 Background

Cross-species protein sequence analysis plays a key role in understanding evolutionary relationships, identifying conserved functional domains, and annotating genes across diverse organisms. One of the most commonly used tools for such analyses is BLASTp (Basic Local alignment for protein), which allows researchers to compare protein sequences and evaluate their similarity [2]. However, performing BLASTp manually using the BLAST software on the NCBI website for large datasets spanning multiple species is a time-consuming and error-prone process, especially when handling numerous protein sequences and generating comparative results.

To address these challenges, this project focuses on developing an automated pipeline for cross-species BLAST analysis. By running BLASTp locally using Python, the pipeline simplifies the workflow, from downloading protein sequences and creating BLAST-compatible databases to running comparative analyses and generating results in CSV format. This automation greatly reduces manual intervention, however, there are a few issues and it can be further optimised.

2 Methods

2.1 Prerequisites and Setup

To ensure the successful execution of the pipeline, the following tools and software were required:

- **Jupyter Notebook:** Used for interactive execution and visualisation of the pipeline.

- **Anaconda:** Provided a Python environment for managing dependencies.
- **BLAST+ Suite:** A toolkit from NCBI for running BLAST queries and creating BLAST-compatible databases [2].
- **Python Libraries:**
  - `pandas`: For data manipulation and handling CSV/Excel files.
  - `Biopython`: For interacting with NCBI databases and handling biological sequence data [1].
  - `openpyxl`: For reading and writing Excel files.
  - `subprocess`: For executing system-level commands from Python.

2.2 Data Preparation

The input data consisted of gene-specific datasets organized in an Excel file (`Gene_data.xlsx`). Each row contained the following information:

- **Gene Name:** The common name of the gene.
- **Gene ID:** A unique identifier for the gene, obtained from the NCBI database.
- **Species:** The species name in which the gene is found.

The Excel file had a sheet named `Gene_names_ids`. The format and structure of the input file were critical to ensuring compatibility with the pipeline. Any discrepancies in the file format could lead to errors during execution.

2.3 Automation Workflow

The pipeline was designed to automate the entire BLASTp workflow. The steps are detailed below:

2.3.1 Library Verification

Before execution, the script verified the installation and functionality of all required Python libraries. Missing libraries were flagged, and users were prompted to install them.

2.3.2 Gene ID Validation

The pipeline validated the gene IDs by querying the NCBI database. Any invalid or unrecognized gene IDs were logged and excluded from further analysis, ensuring the accuracy of subsequent steps.

2.3.3 FASTA File Downloads

Protein sequences corresponding to each gene ID were downloaded from the NCBI database in FASTA format. Two types of directories were created:

- **Grouped Directory:** FASTA files organized by gene name for BLAST database creation.
- **Ungrouped Directory:** All FASTA files are stored individually for external use, such as building phylogenetic trees in MEGA.

2.3.4 BLAST Database Creation

The downloaded FASTA files were converted into BLAST-compatible databases using the `makeblastdb` tool. These databases were stored in a directory structure organized by gene name.

2.3.5 Running BLAST Queries

Cross-species BLASTp queries were executed using the created databases. The user specified a query species, and the pipeline compared its protein sequences against:

- The same species (self-comparison) to ensure database accuracy.
- All other species in the dataset to identify homologous proteins.

The results were saved as CSV files in a structured directory.

3 Results

3.1 Output Structure

The pipeline generated a well-organized output directory, simplifying the navigation and analysis of results. The structure of the output directory is as follows:

```
Auto_BLASTp/
+-- results/
|   +-- blast_results_species_A_vs_all/
|       +-- Gene_1/
|           +-- Species_A_self_comparison.csv
|           +-- Species_A_vs_all.csv
|       +-- Gene_2/
|           +-- Species_A_self_comparison.csv
|           +-- Species_A_vs_all.csv
```

- **Self-Comparison Files:** For each gene, a CSV file (`Species_A_self_comparison.csv`) was generated. This file contained BLASTp results that compare protein sequences within the query species, verifying the integrity of the BLAST database.
- **Cross-Species Comparison Files:** A second CSV file (`Species_A_vs_all.csv`) summarized the BLASTp results of the query species against all other species in the dataset.

Each CSV file contained the following columns:

- **Gene Name:** The name of the gene being analyzed.
- **Query Species:** The species from which the query protein originated.
- **Target Species:** The species being compared to the query.
- **% Identity:** The percentage of identical amino acids in the alignment.
- **Alignment Length:** The number of amino acids aligned between the query and target sequences.
- **Bit Score:** A measure of alignment quality.

4 Discussion

4.1 Performance Evaluation

After testing on 48 proteins across the 7 drosophila species, it was found that it is much faster than using the NCBI BLASTp website but there are still some ways that this pipeline can be optimised.

4.2 Challenges and potential optimisations

Key challenges during development included:

- When preparing the CSV file for input, each specific gene ID for each protein for each species needs to be manually entered in the CSV file. This is quite tedious and time-consuming.
  - To fix this the code should be optimised to be able to search for the fasta file on the NCBI dataset only by using the gene name and species, however, I had some difficulties implementing this.
- When running MEGA to create phylogenetic trees using the **Ungrouped directory** MEGA would often crash simply because of the large number of proteins.
  - To fix this we simply can download the individual fasta files for each protein and species.
- Each gene would have many isomorphisms and as well this BLAST would sometimes analyse multiple sections of the same gene.
  - To fix this a threshold of 70% was implemented and for the e-value a threshold of 0.001. This was implemented since some of these sections were irrelevant. Alternatively, the top alignment percentage could also be chosen for each protein for each species if you did not want to look at isomorphisms.

5 Conclusion

We developed an automated pipeline for cross-species BLASTp analysis simplifying the process of comparing protein sequences across species. Automating tasks such as querying, downloading sequences, and performing comparisons, reduces manual work and improves efficiency, especially for large datasets. Future improvements, such as gene name-based sequence retrieval and further performance optimisation, will enhance its usefulness.

While the pipeline has been tested on a specific dataset of 48 proteins across 7 drosophila species, challenges such as the manual entry of gene IDs, and the presence of isomorphisms highlight areas for further optimisation. Addressing these issues, such as enabling gene name-based sequence retrieval and refining the analysis parameters, will enhance its usability and accuracy.

In conclusion, this pipeline offers a start for developing an optimised software for automating cross-species protein sequence analysis. Future improvements will focus on expanding its capabilities, optimising performance, and validating its application across diverse datasets to better support a wide range of applications.

5.1 Repository containing code and how-to guide

Click here to access the GitHub repository.

References

[1] The Biopython Consortium. Biopython: Bioinformatics tools for python, 2024.

[2] National Center for Biotechnology Information (NCBI). Blast: Basic local alignment search tool, 2024.

[3] P. R. Senthilan, D. Piepenbrock, G. Ovezmyradov, B. Nadrowski, S. Bechstedt, S. Pauls, M. Winkler, W. Möbius, J. Howard, and M. C. Göpfert. Drosophila auditory organ genes and genetic hearing defects. *Cell*, 151(5):1066–1079, 2012.