Querying and analyzing protein data using MongoDB and Neo4j

November 13, 2025

Project Description

Thanks to recent developments in genomic sequencing technologies, the number of protein sequences in public databases is growing enormously. In order to exploit more fully this huge quantity of data, protein sequences need to be annotated with functional properties such as Enzyme Commission (EC) numbers and Gene Ontology terms. The UniProt Knowledgebase (UniProtKB) is currently the largest and most comprehensive resource for protein sequence and annotation data. According to the March 2018 release of UniProtKB, some 568,000 sequences are manually curated but over 229 million sequences lack functional annotations. The ability to automatically annotate protein sequences in UniProtKB/TrEMBL, the non-reviewed UniProt sequence repository, would represent a major step towards bridging the gap between annotated and unannotated protein sequences.

The aim of this project is to first store protein data in a Document store (MongoDB). Then, construct a graph representation of the UniProt protein database (or a part of it!) in which each node of the graph represents maintains some protein-related attributes. These attributes include protein taxonomy, function and sequence, for example. An edge between two nodes means that the linked proteins are similar (e.g. they belong to the same InterPro group) or they share the same protein signatures (e.g. domains and functional sites). Based on one or more attribute of interest, a protein graph may be constructed. The protein graph assigns the reviewed proteins its attribute value. The non-reviewed proteins are un-labeled nodes. The graph representation of the protein data may then be input to a label propagation algorithm that aims to infer attribute values of the unlabeled nodes.

1 Task1: A document store for protein data

MongoDB is one of the leading document databases, a type of NoSQL database with a document-based data model. Documents can contain key-value pairs,

arrays and nested documents (a document of documents). A core use case for a document database is to back search and browsing for a protein database.

The functional requirements here need to support a protein databank with complex querying and filtering across many protein attributes. We need to be able to populate the application view of our protein databank with a single query.

2 Task2: Big graph construction

A common way of building a protein graph is to use the domain composition of protein data. This is a novel way of connecting the proteins using their constituent protein domains. Domains may be considered as natural building blocks of proteins. During evolution, protein domains have been duplicated, fused, and recombined in different ways to produce proteins with distinct structures and functions. Here, each node of the network represents a protein while a link between two nodes means that the proteins exhibit a given minimum level of domain similarity. Thus, each node u is identified by a set of labels $L(u)$ (one or more annotations to propagate), has a set of neighbours $N(u)$, and for every neighbour v it has an associated weight $W_{u,v}$.

In the context of this task, you should provide ideas and implement a bid data solution to construct efficiently the protein graph. A detailed explanation of the proposed solution should be provided.

Domain Based Protein-Protein Network (PPN)

To illustrate the construction of the PPN, let us consider five proteins with symbolic names $P1$,$P2$, $P3$, $P4$ and $P5$. Let us assume that these proteins are composed of domains $D1 = (d1, d2, d3, d4)$, $D2 = (d1, d3, d5)$, $D3 = (d1, d2, d10)$, $D4 = (d5, d6, d1)$, and $D5 = (d4, d1, d10, d40, d7, d9, d12, d52, d100)$, respectively. It is then evident that proteins $P1$ and $P2$ contain two domains in common namely $d1$ and $d3$. Therefore, proteins $P1$ and $P2$ may be linked and the number of shared domains may serve as link weight such as

$$W_{P1,P2} = |(d1, d2, d3, d4) \cap (d1, d3, d5)| = |(d1, d3)| = 2. \quad (1)$$

In a similar way, proteins $P1$ and $P5$ may be linked with a link weight of

$$W_{P1,P5} = |(d1, d2, d3, d4) \cap (d4, d1, d10, d40, d7, d9, d12, d52, d100)| = |(d1, d4)| = 2. \quad (2)$$

In the both cases, the link weight is two. However, the link weight computed in this way does not reflect the true strength of the relationship among the proteins. Because, in the first case, there are total of $|(d1, d2, d3, d4) \cup (d1, d3, d5)| = |(d1, d2, d3, d4, d5)| = 5$ different domains among the two proteins and two are shared. In the second case, there are $|(d1, d2, d3, d4) \cup (d4, d1, d10, d40, d7, d9, d12, d52, d100)| = 11$ different domains of which two are again shared. Although two domains are shared in each case, P1 is intuitively

more aligned with P2 than P5. Therefore, instead of using the above raw similarity score, we instead use the Jaccard index, or Jaccard similarity coefficient, to reflect better the similarity in composition. This is calculated as $\frac{|A \cap B|}{|A \cup B|}$, where A and B are the two sets of constituent domains. Using the Jaccard coefficient, the link weights for P1 and P2 are calculated as:

$$W_{P1,P2} = \frac{|(d1, d2, d3, d4) \cap (d1, d3, d5)|}{|(d1, d2, d3, d4) \cup (d1, d3, d5)|} = \frac{|(d1, d3)|}{|(d1, d2, d3, d4, d5)|} = \frac{2}{5} = 0.4. \quad (3)$$

In other words, according to the Jaccard measure, protein P1 and P2 are 40% similar in their domain composition.

Similarly for P1 and P5, the Jaccard link weight is calculated as

$$W_{P1,P5} = \frac{|(d1, d2, d3, d4) \cap (d4, d1, d10, d40, d7, d9, d12, d52, d100)|}{|(d1, d2, d3, d4) \cup (d4, d1, d10, d40, d7, d9, d12, d52, d100)|} = \frac{2}{11} = 0.18 \quad (4)$$

In this case, P1 and P5 are roughly 18% similar in their domain composition.

The graphs is a weighted un-directed graph. It contains nodes that are labelled and nodes that are un-labelled.

3 Task3: Querying the protein databases

The goal of this task is to propose to the user the possibility to query both protein databases: (1) the document (MongoDB) protein databank and (2) protein graph database (Neo4j, Titan, ...). The main goals are the following:

1. In the document store, search a protein by its identifier and/or name and/or description.
2. In the graph databasse, search a protein/node by its identifier and/or name and/or description. As a result, the user can view the protein, its neighbors and the neighbors of neighbors.
3. Compute some statistics such as the number of labelled and un-labbeled proteins, isolated proteins (protein with no neighbors),
4. Visualize a specific protein (and its neighborhood)

4 Task 4: Protein Function Annotation Task

The goal of this task is to perform a classification/annotation task. The protein network contains few nodes that are labelled with respective function where as a large number of nodes do not have any function annotations. Annotated nodes can be labelled with Terms from Gene Ontology (e.g. GO:10004) or Enzyme commission number (e.g EC 1.3.5.25). This a multi-label classification problem, as each of the node can have more than one labels.

The problem can be seen as link completion or recommendation system where we need to recommend approciate link to connect un-annotated proteins with their respective function or a general multi-label machine learning problem where annotated nodes are the training examples.

5 Project Data

5.1 Small dataset

A pre-processed dataset that contains domain and EC information for Viruses is available in "Moodle".

5.2 Big datasets

Raw data (optional) are accessible from the following sources:

1. ftp://ftp.ebi.ac.uk/pub/databases/interpro/protein2ipr.dat.gz: here, each protein from uniprot is associated with their corresponding domain information in details.
2. ftp://ftp.ebi.ac.uk/pub/databases/GO/goa/UNIPROT/goa_uniprot_all.gaf.gz: this data contains go annotations of each uniprot protein.
3. ftp://ftp.expasy.org/databases/enzyme/enzyme.dat Contains EC informations of Uniprot proteins.
4. http://purl.obolibrary.org/obo/go/go-basic.obo: this file contains the GO Ontology. In fact, to realize the global picture of the annotations and inference, it will be very helpful to have a graph with all the information with these three files.

6 Project Evaluation

The defense

Duration: 20 minutes / 5 minutes for technological choices and 15 minutes for system demonstration

Mark

1. Basic functionalities.

   The creation of two NoSQL databases (document store and graph database) for storing and querying protein data ensure a mark of 10/20. This mark takes into account the quality of the presentation, the motivations of technological choices and the performance of the implemented system. Without basic functionalities, the attributed mark will be 0/20.

2. Additional functionalities:

    - Graphical interface: 3 pt. The system propose an intuitive GUI.
    - Good visualization of protein neighborhood: 2pt
    - Personalized label propagation on protein graphs: 3 pt. This task consists on performing the protein function annotation using several attributes (GO terms, EC numbers, ...).
    - Statistics: 2pt

Libraries

For the visualisation: Neovis, D3JS, ...

Pyhton, JAVA, ...

Angular