SEMANTIC ENRICHMENT AND SIMILARITY APPROXIMATION FOR
BIOMEDICAL SEQUENCE IMAGES

By

Syed Ahmad Chan Bukhari

Master of Engineering, Gyeongsang National University Korea, 2012
Master in Information Technology, University of the Punjab Pakistan, 2008

A Dissertation Submitted in Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy

In the Graduate Academic Unit of Computer Science

Supervisor: Christopher J.O. Baker, Ph.D., Dept. of Computer Science and Applied Statistics

Examining Board: Scott Pavey, Ph.D., Department of Science, Applied Science and Engineering
Bruce Spencer, Ph.D., Faculty of Computer Science
Paul Cook, Ph.D., Faculty of Computer

External Examiner: Gregory Butler, Ph.D., Dept. of Computer Science and Software Engineering, Concordia University

This dissertation is accepted by the Dean of Graduate Studies
April, 2017

THE UNIVERSITY OF NEW BRUNSWICK

© Ahmad Chan, April, 2017
ABSTRACT

Scientific publications are considered as the most up-to-date resource of ongoing research activities and scientific knowledge. Efficient practices for accessing biomedical publications are key to allowing a timely transfer of information from the scientific research community to peer investigators and other healthcare practitioners. Biomedical sequence images published within the literature play a central role in life science discoveries. Whereas advanced text-mining pipelines for information retrieval and knowledge extraction are now commonplace methodologies for processing documents, the ongoing challenges associated with knowledge management and utility operations unique to biomedical image data are only recently gaining recognition. Sequence images depicting key findings of research papers contain rich information derived from a wide range of biomedical experiments. Searching for relevant sequence images is however error prone as images are still opaque to information retrieval and knowledge extraction engines. Specifically, there is no explicit description or annotation of the sequence image content. Moreover, traditional biomedical search engines, which search image captions for relevant keywords only, offer syntactic search mechanisms without regard for the exact meaning of the query. As proposed in this thesis, semantic enrichment of biomedical sequence images is a solution which adopts a combination of technologies to harness the comprehensive information associated with, and contained in, biomedical sequence images. Extracted information from sequence images is used as seed data to aggregate and
harvest new annotations from heterogeneous online biomedical resources. Comprehensive semantic enrichment of biomedical images incorporates a variety of knowledge infrastructure components and services including image feature extraction, semantic web data services, linked open data and crowd annotation.

Together, these resources make it possible to automatically and/or semi-automatically discover and semantically interlink new information in a way that supports semantic search for sequence images. The resulting enriched sequence images are readily reusable based on their semantic annotations and can be made available for use in ad-hoc data integration activities. Furthermore, to support image reuse this thesis introduces a mechanism for identifying similar sequence images based on fuzzy inference and cosine similarity techniques that can retrieve and classify the related sequence images based on their semantic annotations. The outcomes of this research work will be relevant to a variety of user groups ranging from clinicians and researchers searching with sequence image data.
DEDICATION

I dedicate this thesis to my family and my late Grandfather, Syed Baghdad Hussain, who always believed that I could achieve a lot in my life and supported me in difficult times.
ACKNOWLEDGEMENTS

This thesis could not have been finished without the help and support of many people who are gratefully acknowledged here. In the first place, I am honored to express my deepest gratitude to my dedicated supervisor, Dr. Chris J.O. Baker, without whose guidance, kindly concern, and generous support, I could not have worked out this thesis. He always put high priority on my thesis writing and was willing to discuss with me at any time. His patience and kindness are greatly appreciated. Dr. Scott Pavey has offered me valuable ideas, suggestions, and criticisms from his rich research experience. I have learned from him a lot not only about thesis writing, but also professional ethics. I am very much obliged to his efforts in helping me to complete the research. I would like to thank all the other professors in the department of computer science who have been very helpful to me during the course of my study. Thanks to all officers in the University of New Brunswick for their kindness to me. I am also thankful to my close friends Sadnan, Artjom, and Jahandad for their help, support, interest and valuable hints. Finally, I would like to give my special thanks to my dear parents for their encouragement and love that enabled me to complete this work.

February 23rd, 2017

Syed Ahmad Chan Bukhari
Table of Contents

ABSTRACT .............................................................................................................. ii
DEDICATION ........................................................................................................ iv
ACKNOWLEDGEMENTS ..................................................................................... v
Table of Contents ................................................................................................ vi
List of Tables ....................................................................................................... vii
List of Figures ...................................................................................................... viii
List of Symbols, Nomenclature and Abbreviations .............................................. xii
Chapter 1 ............................................................................................................. 1
Introduction ......................................................................................................... 1
1.1 Importance of Biomedical Images ................................................................. 1
1.2 Biomedical Search Engines ......................................................................... 2
    1.2.1 Yale Image Finder (YIF) ...................................................................... 3
1.3 The SEBI Framework .................................................................................. 4
1.4 Thesis Aims ................................................................................................ 7
1.5 Summary of Thesis Contributions ............................................................... 8
    1.5.1 Sequence Image Processing ................................................................. 8
    1.5.2 Sequence Similarity and Categorization ............................................. 9
1.6 Client Interaction with SEBI .................................................................... 11
1.7 Thesis Outline ........................................................................................... 13
Chapter 2 .......................................................................................................... 15
Background ....................................................................................................... 15
2.1 Semantic Web Technologies ....................................................................... 15
    2.1.1 An Overview of Semantic Web ........................................................ 16
    2.1.2 An Overview of Ontology ................................................................. 18
    2.1.3 Components of Ontology ................................................................ 20
    2.1.4 Ontology for Knowledge Representation ........................................ 21
    2.1.5 Ontology Development Steps ......................................................... 23
    2.1.6 Linked Open Data ............................................................................ 24
2.2 Bio-Imaging Ontologies ............................................................................ 28
    2.2.1 DICOM Standard and Ontology ....................................................... 28
    2.2.2 The RadLex Ontology ..................................................................... 29
    2.2.3 The Open Microscopy Environment ............................................. 29
    2.2.4 Quantitative Imaging Biomarker Ontology (QIBO) ....................... 30
    2.2.5 Radiology Gamtus Ontology .......................................................... 30
    2.2.6 The OntoNeuroLog Ontology ......................................................... 31
    2.2.7 The BIM Ontology .......................................................................... 32
2.3 Techniques for Curation and Manipulation of Annotations ....................... 32
Chapter 3: Semantic Sequence Image Processing

3.1 Introduction

3.1.1 Sequence Image Acquisition .......................................................... 63
3.1.2 Optical Character Detection and Extraction ........................................ 63
3.1.3 Overview of OCR Tools and Technologies ........................................... 65

3.2 Methods................................................................................................. 67

3.2.1 Sequence Image Preprocessing .......................................................... 68
3.2.2 System Requirements for OCR Algorithm Training ............................... 71
3.2.3 Training of the OCR Algorithm ......................................................... 72
3.2.4 OCR Post-processing ....................................................................... 76
3.2.5 WordNet and SNOMED CT ............................................................... 77
3.2.6 Global and Local Level Classification of Sequence Images ............... 78

3.3 Results and Evaluation........................................................................ 80

3.3.1 Performance Evaluation of OCR Algorithm ........................................ 81
3.3.2 Measurement of Character Level Accuracy (CLA) ................................ 82
3.3.3 Character Level Accuracy Comparison .............................................. 86

3.4 Discussion (of Sequence Image Processing Outcomes) ......................... 98
List of Tables

Table 2.1: Feature Comparison of SADI with Frameworks……………………………………..41
Table 3.1: Character Level Accuracy with unprocessed images and untrained algorithm………………………………………………………………………………….83
Table 3.2: Character Level Accuracy with pre-processed and with low quality images…84
Table 3.3: Character Level Accuracy with high quality unprocessed images …………..85
Table 3.4: Character Level Accuracy with high quality processed images………………86
Table 3.5: Summary of OCR Algorithm Training and Testing …………………………….89
Table 3.6: Image Signatures for the Datasets used in SEBI OCR Training …………..90
Table 3.7: Image Signatures for the Dataset used in SEBI OCR Testing …………………96
Table 4.1: A SADI Registry of BioNLP and Sequence Analysis Web Services………..106
Table 4.2. Fuzzy Rules Matrix for Sequence Image Categorization …………………….113
Table 4.3: SEBI Performance Evaluation …………………………………………………….118
Table 4.4: SEBI Performance Comparison with Similarity Calculation Algorithms …..119
List of Figures

Figure 1.1: Graphical View of Semantic Enrichment Process of Biomedical Sequence Images ................................................................. 5
Figure 1.2: The SEBI Framework ........................................................................ 6
Figure 1.3: An Integrated View of Domeo with the SEBI SPARQL Endpoint and Yale Image Finder ................................................................. 12
Figure 2.1: The Semantic Web Layered Architecture ........................................ 18
Figure 2.2: Venn Diagram of the Open Data Concept ........................................ 25
Figure 2.3: Venn Diagram Describing Linked Open Data Principle .................... 26
Figure 2.4: Graphical View of Linked Open Data (LOD) Cloud ......................... 27
Figure 2.5: A SADI Service Execution Steps .................................................. 39
Figure 2.6: BioNLP-SADI in Context .............................................................. 40
Figure 2.7: A Graphical View of the Phmmer SADI Service Modeling ............... 43
Figure 2.8: SADI Service Modeling for a Drug Extraction Service .................... 44
Figure 2.9: Drug-Extraction and Drug-Drug-Interaction SADI Services Output Modeling ................................................................. 45
Figure 2.10: An Example of a Radiograph ....................................................... 48
Figure 2.11: Functional MRI of Brain Activities .............................................. 49
Figure 2.12: A view of Fluorescent Microscopy Inside a Cell ............................ 50
Figure 2.13: Histogram of an Organism .......................................................... 51
Figure 2.14: A Labeled Gel Electrophoresis Image .......................................... 51
Figure 2.15: An Example of a Protein Sequence Image found in Biomedical Literature ................................................................. 53
Figure 2.16: An Example of a Multiple Sequence Alignment ............................. 54
Figure 2.17: Fuzzy Membership Function ...................................................... 60
Figure 2.18: Fuzzy Rule Engine General Architecture ...................................... 61
Figure 3.1: Multiple Sequence Extraction from an Image .................................. 65
Figure 3.2: Low Resolution Image from Yale Image Finder ................................ 68
Figure 3.3: An Image Post-resolution and Gray Scale Filter Adjustment ............ 69
Figure 3.4: A Sequence Image after the Application of a Gaussian Filter .......... 70
Figure 3.5: A Sequence Image View after Laplace Filter’s Application .............. 71
Figure 3.6: A View of Sequence Image with Text Surrounded Histograms……………71
Figure 3.7: The jTessBoxEditor view with Input File (Training Step -1) ………………73
Figure 3.8: A Component Analysis View of Training File in jTessBoxEditor ………74
Figure 3.9: Box File Editing Process with jTessBoxEditor………………………………75
Figure 3.10: A Flow Chart of High Resolution Image Acquisition from PubMed API..76
Figure 3.11: The DNA and Protein Sequence Segregation Algorithm…………………77
Figure 3.12: Comparison of Character Level Accuracy……………………………….87
Figure 3.13: A Graphical Comparison of Job Completion Time .……………………88
Figure 4.1: The Graphical View of Sequence Image Similarity in SEBI Context …..102
Figure 4.2: BIM Modeling to convert the Sequence Analysis Annotations into RDF...109
Figure 4.3 A Vector Space Plot Illustrating the Cosine Similarity……………………..111
Figure 4.4: A Graphical Comparison Among Similarity Measures Using SEBI ……120
Figure 4.5: An Illustration of Queries Execution Time Comparison…………………..121
List of Symbols, Nomenclature and Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>BIM</td>
<td>Biomedical Image Ontology</td>
</tr>
<tr>
<td>BLAST</td>
<td>Basic Local Alignment Search Tool</td>
</tr>
<tr>
<td>CBIR</td>
<td>Content-based Image Retrieval</td>
</tr>
<tr>
<td>CDD</td>
<td>Conserved Domain Database</td>
</tr>
<tr>
<td>CLA</td>
<td>Character Level Accuracy</td>
</tr>
<tr>
<td>CMF</td>
<td>Concept Matching Frequency</td>
</tr>
<tr>
<td>CSM</td>
<td>Cosine Similarity Measure</td>
</tr>
<tr>
<td>CT</td>
<td>Computed Tomography</td>
</tr>
<tr>
<td>CUDA</td>
<td>Compute Unified Device Architecture</td>
</tr>
<tr>
<td>DAML</td>
<td>DARPA Agent Markup Language</td>
</tr>
<tr>
<td>DARPA</td>
<td>Defense Advanced Research Projects Agency</td>
</tr>
<tr>
<td>DIA</td>
<td>Deep Image Analysis</td>
</tr>
<tr>
<td>DICOM</td>
<td>Digital Imaging and Communications in Medicine</td>
</tr>
<tr>
<td>DNA</td>
<td>DeoxyriboNucleic Acid</td>
</tr>
<tr>
<td>DOLCE</td>
<td>Descriptive Ontology for Linguistic and Cognitive Engineering</td>
</tr>
<tr>
<td>DOMEO</td>
<td>Document Metadata Exchange Organizer</td>
</tr>
<tr>
<td>EMBL</td>
<td>European Molecular Biology Laboratory</td>
</tr>
<tr>
<td>FASTA</td>
<td>Fast Adaptive Shrinkage Thresholding Algorithm</td>
</tr>
<tr>
<td>FIE</td>
<td>Fuzzy Inference Engine</td>
</tr>
<tr>
<td>Acronym</td>
<td>Full Form</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>FMRI</td>
<td>Functional Magnetic Resonance Imaging</td>
</tr>
<tr>
<td>GAs</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>GIF</td>
<td>Graphics Interchange Format</td>
</tr>
<tr>
<td>GO</td>
<td>Gene Ontology</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HP</td>
<td>Hewlett Packard</td>
</tr>
<tr>
<td>HITS</td>
<td>High Identity with Tolerance.</td>
</tr>
<tr>
<td>HMMER</td>
<td>Hidden Markov Model-based sequence alignment tool</td>
</tr>
<tr>
<td>HSL</td>
<td>Hue-saturation-lightness</td>
</tr>
<tr>
<td>HSV</td>
<td>Hue-saturation-value</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>IAA</td>
<td>Image Acquisition Algorithm</td>
</tr>
<tr>
<td>IDM</td>
<td>Image data manipulation</td>
</tr>
<tr>
<td>IME</td>
<td>Image manipulation engine</td>
</tr>
<tr>
<td>JCT</td>
<td>Job completion time</td>
</tr>
<tr>
<td>JPEG</td>
<td>Joint Photographic Experts Group</td>
</tr>
<tr>
<td>JSRT</td>
<td>Japanese Society of Radiological Technology</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine</td>
</tr>
<tr>
<td>KEGG</td>
<td>Kyoto Encyclopedia of Genes and Genomes</td>
</tr>
<tr>
<td>LDM</td>
<td>Linked Data Manager</td>
</tr>
<tr>
<td>LOD</td>
<td>Linked Open Data</td>
</tr>
<tr>
<td>MRI</td>
<td>Magnetic Resonance Imaging</td>
</tr>
<tr>
<td>NBIA</td>
<td>National Biomedical Imaging Archive</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>NLM</td>
<td>National Library of Medicine</td>
</tr>
<tr>
<td>NLP</td>
<td>Natural Language Processing</td>
</tr>
<tr>
<td>OCT</td>
<td>Optimal Cutting Temperature</td>
</tr>
<tr>
<td>OIL</td>
<td>Ontology Inference Layer or Ontology Interchange Language</td>
</tr>
<tr>
<td>OME</td>
<td>Open Microscopy Environment</td>
</tr>
<tr>
<td>ONL</td>
<td>OntoNeuroLog Ontology</td>
</tr>
<tr>
<td>OGD</td>
<td>Open Government Data</td>
</tr>
<tr>
<td>OCR</td>
<td>Optical Character Recognition</td>
</tr>
<tr>
<td>OWL</td>
<td>Web Ontology Language.</td>
</tr>
<tr>
<td>PAL</td>
<td>Protégé Axiom Language</td>
</tr>
<tr>
<td>PET</td>
<td>Positron Emission Tomography</td>
</tr>
<tr>
<td>PHIL</td>
<td>Public Health Image Library</td>
</tr>
<tr>
<td>PM</td>
<td>PubMed</td>
</tr>
<tr>
<td>PMC</td>
<td>PubMed Central</td>
</tr>
<tr>
<td>QET</td>
<td>Query Execution Time</td>
</tr>
<tr>
<td>QIBA</td>
<td>Quantitative Imaging Biomarker Alliance</td>
</tr>
<tr>
<td>QIBO</td>
<td>Quantitative Imaging Biomarker Ontology</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>REST</td>
<td>REpresentational State Transfer</td>
</tr>
<tr>
<td>RGO</td>
<td>Radiology Gamtus Ontology</td>
</tr>
<tr>
<td>RNA</td>
<td>Ribonucleic Acid</td>
</tr>
<tr>
<td>ROI</td>
<td>Region of Interest</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>RSNA</td>
<td>Radiological Society of North America</td>
</tr>
<tr>
<td>SADI</td>
<td>Semantic Automatic Discovery and Integration</td>
</tr>
<tr>
<td>SEBI</td>
<td>Semantic Enrichment of biomedical sequence Images</td>
</tr>
<tr>
<td>SEDI</td>
<td>Semantic DICOM ontology</td>
</tr>
<tr>
<td>SHARE</td>
<td>Semantic Health And Research Environment</td>
</tr>
<tr>
<td>SISA</td>
<td>Sequence Image Segregation</td>
</tr>
<tr>
<td>SNOMED</td>
<td>Systematised Nomenclature of Medicine</td>
</tr>
<tr>
<td>SPARQL</td>
<td>SPARQL Protocol and RDF Query Language</td>
</tr>
<tr>
<td>SPECT</td>
<td>Single-Photon Emission Computed Tomography</td>
</tr>
<tr>
<td>SSWAP</td>
<td>Simple Semantic Web Architecture and Protocol</td>
</tr>
<tr>
<td>SPC</td>
<td>Stream Processing Core</td>
</tr>
<tr>
<td>TIFF</td>
<td>Tagged Image File Format</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>VSM</td>
<td>Vector Space Model</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>WSDL</td>
<td>Web Services Description Language</td>
</tr>
<tr>
<td>WSMO</td>
<td>Web Service Modeling Ontology</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Importance of Biomedical Images

During the last two decades, scientific experiments have produced enormous amounts of data, now known as “big data”, due to the advancement of high-throughput technologies. Efficient practices for providing access to biomedical publications are essential in facilitating the timely transfer of information from the scientific research community to peer investigators and other healthcare practitioners. Biomedical sequence images are a major component of scientific publications and play a central role in communicating life science discoveries. Ongoing challenges associated with knowledge management and utility operations unique to biomedical image data are only recently gaining recognition.

Images depict key findings of research papers, such as: specific protein-protein interactions (Western blot) [1], the signaling flow between a group of proteins (pathway diagrams) [2], or the location of specific brain activity (functional MRI image) [3] and protein or DNA sequence
alignment images [4]. Therefore, making biomedical image content explicit is essential with regards to medical decisions such as diagnosis, treatment, data management and data secondary use for biomedical research.

1.2 Biomedical Search Engines

Available biomedical archives [5] such as National Biomedical Imaging Archive\(^1\) (NBIA), NIH Images\(^2\), NCI Visuals Online\(^3\), CDC Public Health Image Library (PHIL)\(^4\), Japanese Society of Radiological Technology (JSRT) Database and University of Iowa - Hardin MD\(^5\) are examples of resources that support access biomedical information and permit search for biomedical images. These biomedical search engines predominantly provide a syntactic keyword-based search mechanism that poses a significant limitation to the biomedical community when seeking access to relevant information. In the life sciences, spreadsheets, databases and XML files continue to be the conventional formats used to store experimental data such as: Biota [6], DrugBank [7], and Open Microscopy Environment (OME) [8]. Some projects go further and make their XML data accessible via knowledge-based grid services, e.g. high-throughput biological imaging data [9], and PLAZi [10]. The primary purpose of archiving scientific data is to make it available for reuse and mining for novel and significant patterns within the datasets and to further elaborate a scientific phenomenon that has not been completely understood yet. However, the fact that data is persisted only in legacy formats frequently impedes data integration and scientific discovery. For instance, the tables in a relational database have fixed number of columns and any addition to the database requires basic schema alteration [11]. Moreover, migration of data from one database to another

---

database is a time-consuming and laborious task, as both the databases and XML data files strictly follow their respective schemas. Conventional database systems discourage the integration with other datasets, and yet a generic life science data query often requires information that resides in one or more heterogeneous data resources [12].

### 1.2.1 Yale Image Finder (YIF)

This section introduces (YIF) Yale Image Finder [13] which is a reliable and widely used biomedical image search engine. Biomedical sequence image datasets have been acquired from YIF for the SEBI experiments. YIF permits retrieval of biomedical images and associated data based on queries made over the metadata of the images. YIF repository currently holds over 1.5 million biomedical images and associated metadata in its index. The data held in YIF comes from open source PubMed articles under license from the NLM\(^6\) (National Library of Medicine) as XML data files. All relevant metadata is extracted (e.g. title, authors, PM and PMC IDs, and year) for each article, and stored in a Lucene\(^7\) index known for rapid information retrieval capabilities. However, there are several technical issues that are associated with such kind of image management solutions. Specifically, the directory structure, where each directory behaves as a separate entity, does not synchronize itself with ongoing changes that might occur in other directories. Therefore, data redundancies and delete or update anomalies may occur. In the current implementation, YIF identifies two classes of images: axis diagrams and gels [14], which are classified using the segmentation results and frequently-used image features such as Haralick features, grayscale histogram features, along with text features from image captions.

\(^6\) https://www.nlm.nih.gov/

\(^7\) https://lucene.apache.org/core/
1.3 The SEBI Framework

Biomedical sequence images depict key findings of research papers. Searching for images using available biomedical image search engines [5] is an error-prone activity since the biomedical sequence images are still opaque to knowledge extraction engines. In general, the legacy data management techniques, image manipulation and the unavailability of explicit descriptions about sequence image contents are the factors that impede the traditional biomedical search engines. Syntactic keyword-based search mechanisms constitute a significant limitation to the biomedical community in achieving important information retrieval activities.

In order to resolve the findability, accessibility, interoperability, reusability and provenance challenges faced by the current biomedical images portal, this thesis presents a general framework for the semantic enrichment and the similarity approximation of biomedical sequence images called SEBI [15]. SEBI utilizes the information (optical characters) extracted from sequence images as seed data to harvest new annotations from heterogeneous online biomedical resources and measures the similarity of sequence images based on their derived annotations. The SEBI framework essentially incorporates image analysis [16], biological text mining [17], sequence analysis [18], semantic web data services [19], linked image data [20], similarity measurement algorithms [21] and crowd annotation methodologies to automatically and semi automatically generate semantic annotations on the sequence images to support sequence image search and extraction of related information. Sections 4.1.1 and 4.1.2 provide more detail about SEBI sequence similarity algorithm and categorization techniques. Figure 1.1 displays a semantic enrichment process specific to biomedical sequence images which provides machine process-able annotations.
Semantic enrichment of sequence images is a series of activities as displayed in Figure 1.1 with labels from 1 to 6. The semantic enrichment process starts (Label 1) with sequence image acquisition followed by optical character recognition (Label 2). Optical characters (sequence strings) are passed (Label 3) to various web services to generate annotations. The generated annotations are converted to RDF format (Label 4) and stored as a consolidated RDF graph (Label 5) in triplestore (Label 6). In the SEBI framework, semantic enrichment and similarity approximation is performed at different stages. The SEBI framework can be described in four tiers as displayed in Figure 1.2 namely: image data manipulation (IDM), image annotation generation, semantic enrichment chamber and user interaction tiers. Figure 1.1 is a process diagram and provides a step-by-step description of sequence image enrichment in SEBI while Figure 1.2 illustrates the overall SEBI framework. SEBI follows a multi-tier system design where each tier performs a number of subtasks as summarized below.
Tier 1 carries out image data manipulation introducing the IAA (Image Acquisition Algorithm) that acquires the images from YIF, removes redundant copies, and adds any missing metadata information drawing updates via the PubMed API [22]. The IME (Image Manipulation Engine) at Tier 2 performs preprocessing to enhance sequence image resolution, contrast and color scheme by employing a number of image filters (Section 3.2.1). For biomedical sequence image processing and segregation, we have developed DIA (Deep Image Analysis) and SISA (Sequence Image Segregation) algorithms at Tier 2. DIA and SISA algorithms identify the sequence images in a collection of biomedical images (see at Section 2.5) and further segregates the protein and DNA sequence images respectively. These algorithms depend upon an optical character recognition (OCR) engine (Tier 2) which is designed, trained and tested to extract the ASCII characters (sequence strings) from the sequence images (section 3.2.4). OCR uses the artificial intelligence techniques to recognize characters in an image. Post-processing (Label 2 in Figure 1.1) of the
sequence ASCII is also carried out at the IME level to remove the dictionary words, medical terms, special characters and digits to generate a clean sequence string for further analysis. Sequence analysis and BioNLP web services [12] are run to produce the annotations from sequence analysis software such as BLAST, HMMER, Prosite and CDD (Tier 2). Tier 3 introduces the BIM ontology [23] and linked data manager which utilizes semantic modeling to convert the sequence annotations to RDF. BIM ontology is an open source biomedical image ontology that has been designed for SEBI project to provide the relevant semantic vocabularies to convert annotations into RDF. The generated annotations are exposed as linked data and stored in iCyrus [24]. Tier 4 incorporates sequence similarity algorithms and a categorization technique to score a similarity sequence between images based on their derived (newly generated annotations) annotations.

1.4 Thesis Aims

As discussed in sections 1.2 and 1.2.1, there has been quite some work on biomedical images. However, there are still several open problems that remain unexplored, or not completely investigated in retrieval and reuse of biomedical sequence images. This thesis proposes to introduce novel methodologies and infrastructure in support of semantic image search. Key challenges must be overcome to ensure that images can be discovered using domain specific terminologies. Images must have semantic metadata that describe contents of the images. In most cases, it is difficult to summarize an image contents automatically without manual intervention. In the particular case of images in this study, namely sequence images, there is a possibility to process the images and extract strings of characters that can be used to derive additional information about the image. Methodologies to achieve this goal have not yet been explored in detail. The initial hurdle that must be overcome are the accurate recognition and extraction of characters in poor quality images for reuse. One of the aims of this thesis was to develop an
efficient OCR extraction algorithm supporting the generation of image annotations through a data transformation process. Using high quality outputs from OCR, a methodology must be put in place to generate appropriate annotations. Since strings of optical characters from sequence images derived from scientific literature have biological significance, they can be checked against databases of known biological sequences for similarity. Annotations can be gathered from high similarity sequences and transferred to the character strings derived from the images. Using this approach new annotations can be applied to images making them-semantically searchable based on information about the image contents, such as fungal origin or protein property. This thesis aims to explore the feasibility of this approach given the available state of the art resources for OCR and biological sequence analysis. Additional challenges and opportunities must be investigated, specifically key infrastructure for authoring annotations to images must be designed and tested. Further to this, opportunities to develop and similarity algorithms will be tested. This thesis aims to document the extent to which semantic enrichment and similarity approximation can support information retrieval and knowledge discovery for scientists.

1.5 Summary of Thesis Contributions

This section provides a summary of two main contributions that have been addressed by this thesis.

1.5.1 Sequence Image Processing

The processing of sequence images involves image acquisition, preprocessing, training and testing of OCR algorithms, OCR extraction, evaluation of the OCR engine and post processing. The desired goal of sequence image processing is to get the most accurate and complete optical characters out of sequence images. Sequence image processing is fraught with difficulties and
there is no straightforward way to get accurate optical characters from the sequence images with mixed resolutions. Therefore, we developed algorithms i.e. IAA, DIA and SISA, trained and tested the OCR algorithms (Section 3.2.4 and 3.3.1), examined their accuracy and performed the post processing. Acquisition of sequence images from YIF repository is carried out using IAA which is also responsible for detecting and removing any redundant image data and for gathering the missing sequence image metadata through PubMed API [22]. An Image manipulation engine enhances the contrast, resolution, dynamic range and color scheme by recruiting the image filters (Section 3.2.1) prior to submitting them to the OCR algorithm. To increase the accuracy and reduce the response time at the level of optical character extraction, a comprehensive training is required. Therefore, the SEBI OCR algorithm was trained (Section 3.2.2) and tested against datasets with different sequence image resolutions (Section 3.3.1). Moreover, the accuracy of SEBI OCR algorithm was calculated at character level and recorded along with the changes in accuracy and the algorithm response time with font-specific training (Sections 3.2.1 and 3.2.3). The sequence analysis process is not needed for medical terms, dictionary words and special characters. Therefore, we have removed them from the sequence strings (Section 3.2.4) while conducting the post processing. Details of the sequence image processing can be studied in chapter 3.

1.5.2 Sequence Similarity and Categorization

Our efforts at the sequence image similarity and categorization levels include the introduction of a semantic enrichment process, semantic modeling through BIM ontology, image linked data generation, measuring of the semantic similarity of sequence images based on their derived annotations and performing the categorization. The semantic enrichment process involves automatic annotation generation using heterogeneous biomedical resources. We developed a library of biological sequence analysis and BioNLP web services by exposing the sequence
analysis algorithms and BioNLP software i.e. BLAST, HMMER, Prosite and WHATIZIT as web services (Section 4.2.1). BioNLP web services are used to annotate a biological sequence image by extracting the information from the image-associated text (e.g. caption, description) and from original the publication and tagging the image. To measure the similarity of images the generated annotations should be integrated, therefore we developed the BIM ontology that provides common vocabularies to model the annotations (Sections 2.2 and 4.3). We particularly focused on the provenance of generated annotation and sequence images through the introduction of ontology classes and properties in BIM ontology. All image annotations are stored in RDF and a vector space model approach was applied to measure the semantic similarity of sequence images based on their derived annotations (Section 4.2.3). It is noteworthy that sequence images are accessed from PubMed publications therefore the derived annotations include the image provenance information as an additional metadata. To categorize the sequence images based on their degree of membership to a category, we introduced fuzzy-based rules and applied the Fuzzy C-means based categorization (Section 4.2.4). Moreover, we evaluated the system through use case queries and by applying the four similarity measurement algorithms for cross comparison.
1.6 Client Interaction with SEBI

The absence of application programming interfaces in current systems forces the users to study the data after manual downloading it (in the form of a data dump, without metadata and provenance information) from the biomedical imaging portals. The outputs of semantic enrichment of biomedical images are stored in a semantic triple store called iCyrus which is a core component of SEBI framework. iCyrus behaves as a semantic API and is accessible through multiple semantic query mechanisms that include SPARQL queries or faceted browsing to the iCyrus endpoint, third party semantic clients such as DOMEO (which is a web-based annotation generation portal) and SADI federated query clients such as SHARE [25].

DOMEO (Document Metadata Exchange Organizer) [26], is based on Google Web Toolkit and JavaScript. A web browser like feature of DOMEO fetches the webpage and allows to an individual or a group to generate annotations on HTML or XML documents. Annotations stored in iCyrus are available in RDF modeled according to Annotation Ontology or the Open Annotation format [27]. DOMEO was built to operate with any web page and provides support for PubMed Central content. More precisely, when a PubMed Central document is loaded in the annotator, a dedicated software extraction pipeline is triggered and proceeds to extract and retrieve all of the available bibliographic data. Additionally, an index of all of the figures in the document is created. To access iCyrus a server-side plugin for DOMEO was developed. The plugin allows end users to launch defined SPARQL queries from the DOMEO web interface using point and click browser button. These queries select annotations for display and related images to those already shown in the browser from iCyrus and YIF respectively.
DOMEO users can also author manual annotation to publication images for storage in iCyrus. Figure 1. shows the DOMEO plugin for iCyrus. Without connecting to iCyrus images are highlighted in yellow, however, after running the SPARQL queries additional annotations and images are presented.
1.7 Thesis Outline

This thesis introduces the SEBI framework, to enrich biological sequence images with related semantic annotations to make them discoverable. The proposed SEBI framework incorporates a number of technologies to support semantic search for sequence images. These include image OCR extraction, semantic web services, linked data, text mining and algorithms for similarity calculation. Enriched linked image data can be accessed using variety of query clients. The following thesis outline provides more detail on the topics covered in rest of the thesis.

- Chapter 2 provides a brief background on the tools and technologies that have been recruited in the proposed SEBI framework. It explains core components of the semantic web and ontologies and advocates for the adoption of semantic technologies to support knowledge discovery. It outlines the ontology design techniques along with current best practices in semantic knowledge representation for use in linked data and semantic web services. Chapter 2 specifically recommends the use of SADI semantic web services to achieve interoperability of biological sequence analysis and BioNLP services for the ad-hoc semantic consolidation of their combined outputs. It further illustrates the available biomedical image types and outlines techniques used to capture and process biomedical images such that quantitative information can be derived for therapeutic and diagnostic purposes. Particularly, it introduces biological sequence images of DNA and proteins that are found in scientific publications. The last section of Chapter 2 provides a brief overview of biological sequence analysis software and an overview of fuzzy systems for categorization purposes.

- Chapter 3 illustrates biological sequence image processing designed in this thesis to support semantic image enrichment. It consists of several modules mentioned in Tier 1 and Tier 2 of
SEBI framework (see Figure 1.2). Tier 1 summarizes the process of acquisition, preprocessing and initial storage of images. The preprocessing section focuses on the image filters, such as Laplace and Gaussian, that have been employed to improve image resolution. Tier 2 illustrates the image categorization process in detail. It introduces two algorithms that classify an image based on its type (sequence and non-sequence images). To examine the character level accuracy of the OCR engine, Chapter 3 applies the character level accuracy measurement criteria [28] and documents the results.

- Chapter 4 addresses the semantic similarity and categorization concepts and explains how they are applied in the context of our research work. In SEBI, derived annotations that are generated on the sequence image are used to compute the similarity between sequence images. The similarity criteria in SEBI are based on the semantic matches among the derived features provided as semantic annotations rather than the physical features of an image such as texture, color, and pixel. Chapter 4 also explains the process of modeling the web services that accept biological sequence strings as inputs and generate the relevant annotations by calling the sequence analysis software. Furthermore, Chapter 4 explains fuzzy-based soft clustering to categorize annotated sequence images. The BIM ontology is introduced for the purpose of semantically modeling sequence image annotations. Additionally, the performance evaluation of the SEBI image annotation and similarity calculation is described using Precision, Recall and F-measure.

- Chapter 5 summarizes the thesis major contributions followed by the thesis limitations and concludes itself with a future work section.
Chapter 2

Background

This chapter has been divided into seven sections: semantic web technologies, biomedical image ontologies, annotation curation and manipulation, semantic web services and BioNLP, an introduction to biomedical images, sequence analysis tools and fuzzy systems. Each section provides an introduction and a brief literature review of the related topic. The basics that have been provided in this chapter will help readers to better understand this dissertation.

2.1 Semantic Web Technologies

The semantic web technology section introduces semantic web layered architecture and provides detail about ontology evolution, components of ontology, languages and procedure used in ontology development and linked open data generation.
2.1.1 An Overview of Semantic Web

The heterogeneity of data on the world wide web (WWW) is growing rapidly. Thousands of web pages are added to the WWW and millions of transactions take place using e-commerce applications every day [29]. Moreover, the dynamicity in web applications is increasing such that information extraction using conventional technologies from this large pool of data on the WWW is a challenge for the conventional search algorithms. As a result, there has been a strong impetus in the field of information engineering to develop and improve techniques to extract precise information from huge volumes of extraneous data. Currently, the web is no more than an accretion of millions of web pages, whereas the current information extraction tools use syntactic search mechanisms. The conventional search engine is not able to find exact information according to user requirements. Primarily, search engines can be divided into two categories: crawler-based search engines, and directory-based search engines [30]. The crawler-based search engines, such as Google and Yahoo, automatically index web pages that are uploaded on the Internet. Crawler-based search engines collect and index the web pages after applying optimization criteria that vary from search engine to search engine [31]. When a search request is made through a crawler-based search engine, the underlying search algorithms compare keywords against the index of repository pages and display the results that meet the criteria. In the case of directory-based search engines, the users manually add web pages and their corresponding meta-data on the web. Therefore, it is necessary to keep updating the index information to reflect changes in the websites. Crawler-based search engines, however, keep updating their indexes without human intervention as keyword-based search engines extract results based on their matching keywords. Consequently, a high volume of data is recalled, albeit the ratio of precision remains relatively low [30, 31].
When the Internet was invented, the idea was to create a virtual world in which human and computer could work together while sharing information [32]. Although not explicitly stated, the vision of the virtual world implied the development of computer algorithms with the same ability to understand web contents as a human. The continuous unchecked growth of Internet derailed the Internet configuration from its vision. Additionally, at the beginning of the WWW, there was not sufficient impetus to correct the deficiencies regarding relevant and correct information retrieval [32, 33] because of the low volume of data. The current web has been designed for human consumption, it does not provide support for automatic machine interpretation. Computers do not have the ability to understand the actual meaning of sentences as humans do. Computers cannot even differentiate among fairly simple sentences such as: “A man eats a chicken” or “A chicken eats a man” with less sophisticated artificial intelligence algorithms. Several statistical and natural languages processing techniques have been employed in the past to make computers operate intelligently. With natural languages processing, a computer can recognize a sentence syntactically by extracting the nouns, verbs and other part of speech and classify them appropriately. However, a machine still cannot infer new knowledge by analyzing the available data. Web mining techniques have also been unsuccessful because of the volume of data on the Internet [30].

In [34] researchers presented a vision of the web in which the current link directory structures are transformed to large, but efficiently distributed repositories of knowledge. This can be done by adding the semantic annotation to a webpage that creates the ability for machines to understand the underlying meaning of concepts and their relationships. The layered architecture of the semantic web divides the whole process into steps from Unicode/URI to Trust (see Figure 2.1). To add semantic annotations, different techniques were developed; researchers started with XML code that adds metadata to facilitate understanding of web contents. Later, the Resource
Description Framework (RDF) [35, 36] was used to describe web contents. RDF arranges the contents and concepts into triples.

RDF is currently used in many industrial and professional software tools. To expedite the inference process in semantic web, several new techniques, such as ontologies, web rules and digital signatures, have been introduced. Figure 2.1 displays the semantic web layered architecture.

![Figure 2.1: The Semantic Web Layered Architecture [35]](image)

### 2.1.2 An Overview of Ontology

The term ontology has definitions both in philosophical and information science domains. In philosophy, ontology is primarily defined as a systematic explanation of being. In the early 5th century, ontology was introduced as an effective methodology to describe the phenomenon of essence (the roots of something) and existence (the presence of something in the real world) [33]. Moreover, a number of philosophers of that time presented proposals to address the issue of the essence of things during their transition phase. For instance, Parmenides believed in the independence of the essence of things from its connection to our senses. He presented his theory about the independence of essence by taking a common example of a seed and tree. What could
be the essence of a tree and a seed when both are in the growing stages and had already lost their original shapes? Parmenides raised questions. Similarly, Aristotle, the author of Metaphysics, provided the answer of the essence of things in transition by introducing his “Mode of Being” theory. According to Aristotle, it is not the case that something that is not a tree becomes a tree, but essentially a tree changes its mode of being from not completely fulfilled to completely fulfilled [37]. Furthermore, Aristotle introduced some categories such as substance, quality, quantity, relation, action, passion, place and time, to segregate and to leverage the classification process among different modes of a thing’s transition. Kant also presented a categorization scheme, one that was based on quantity, quality, relation and modality attributes. Kant argued that the phenomenon of the essence of things could also be understood with the involvement of whoever perceives and understands things in addition to the things themselves [37].

During the 18th and 19th centuries Gasset and William extended Kant’s work [38] and redefined the “Essence of Things” theory according to an information system point of view. Consequently, in the 19th and 20th centuries, efforts were directed towards defining ontology in formal way. Nino Cocchiarella proposed formal ontology as a systematic workflow in a formal, axiomatic development of the logic of all forms and modes of being [38]. During the past two decades, researchers, have particularly focused on the knowledge engineering domain due to its importance and have sought to redefine ontology as a knowledge engineering tool. Neches and colleagues proposed in 1991 what is considered a pioneering definition of ontology in the knowledge engineering domain. They described ontology as the basic terms and relations comprising the vocabularies of a topic area, as well as the rules for combining terms and relations to define extensions to the vocabulary [39].
This definition highlights the anatomy of ontology by defining its parts. We deduce from this definition that ontology covers domain knowledge by incorporating its components and by creating links between the components. Similarly, Gruber and Borst came up with their own definitions about ontology. Gruber said that ontology is an explicit specification of a conceptualization [40] while Borst extended Gruber's definition to promote its application in computers and declared that ontologies are defined as a formal specification of a shared conceptualization [41]. Borst emphasized the machine readability of an ontology and its collaborative usage. On the other hand, Gruber's definition satisfied the generic concept of an ontology. Guarino and Giaretta in 1995 merged both the definitions stated above to make the definition of ontology more valuable. Guarino and Giaretta perceived ontology as philosophical, informal conceptual, formal semantic specification of a conceptualization and the vocabularies used by a logical theory. They further defined ontology as a logical theory that gives an explicit, partial account of a conceptualization [38, 42]. Comparing the definitions of ontology from the Greek era until now, we can observe a continuous evaluation of the concept with time.

### 2.1.3 Components of Ontology

Ontology comprises a number of components such as: classes, relations, individuals, functions, restriction, rules and axioms. This section covers the important components of ontology [43]. Classes or concepts are taken in a broad sense and regarded as a base component of ontology. Classes in ontology are arranged in taxonomic order and form a hierarchical structure. For example, in the e-health domain, Patients, Hospitals, Beds, Cardiac-patient, Patients’ Wards are the classes, the ontology arranges these classes in a subclass and superclass hierarchy. Therefore, we can classify the Cardiac-Patient and the Kidney-Patient as subclasses of the Patient class. In addition, it is also possible a class could be a union or intersection of subclasses. Relations or
relationships principally play a binding role in ontology modeling. Relationships associate concepts with each other through a binary relation. Moreover, the relations can also link concepts with individuals of the ontology. The most common binary relation is \textit{isSubClassOf} relation \cite{43}. We can demonstrate it as, Cardiac-Patient \textit{isSubClassOf} Patient class. Here the binary relation \textit{isSubClassOf} is being used to build the taxonomy of the ontology. Individuals are also known as instances, where instances are technically the ground level components of an ontology. Instances are used to represent the examples of a class in the ontology. Thus, instances can be any kind of entities like customer, product, and place. For instance, \textit{Saint-John-hospital} and \textit{Ahmad} are individuals of the Hospital and Patient classes respectively. Constants are numerical values which remain static during the entire development process of a domain ontology, such as 37 centigrade is the normal body temperature and it remains constant in the health knowledge system development process \cite{37, 43}. Functions are a special sort of relation; for example, Patient-\textit{Fee} is a fee which is charged after the deduction of insurance. Formal axioms are logical expressions and serve the purpose of defining the constraints in an ontology. Axioms translate a generic ontology to a domain specific one representing the domain specific knowledge using specific language such as PAL (Protégé Axiom Language) \cite{44}. Similarly, rules are used with ontologies to increase the expressivity of the domain knowledge so that it is possible to infer the implicit knowledge.

\textbf{2.1.4 Ontology for Knowledge Representation}

Relevant information extraction and interpretation are the major challenges that researchers are facing nowadays. The inaccessible part of the web is considered to be five hundred times more than what available search engines can find \cite{5}. Semantic web (see section 2.1) enables web-based applications to retrieve the relevant information intuitively and make intelligent decisions. The key innovation is making web content process-able for computers. Markup languages such as XML
offer a set of rules to describe the structure of web documents that allows machines to process the information at a certain level. However, because of document-specific schema, XML-based documents cannot support semantic interoperability. The semantics of the data should be completely comprehensible by the machines for context-aware information discovery. Semantic web introduces the Resource Description Framework (RDF) [36], a W3C-endorsed framework to represent data. RDF represents data in a subject, predicate, object triple format such that Ahmad is a Student. Here, Ahmad is a subject, “is a” is the predicate, and student is an object. Moreover, RDF can represent the application-specific classes and properties with RDF schema. By itself, RDF is just a data model; it does not have any significant semantics.

RDF schema provides a vocabulary to describe the classes of the domain. For example, it provides `rdfs:Class`, `rdfs:Property` (from the RDF namespace), `rdfs:subClassOf`, `rdfs:subPropertyOf`, `rdfs:domain`, and `rdfs:range`. It also provides properties for metadata, such as `rdfs:label` and `rdfs:comment`. RDF schema is considered a weak candidate in semantic primitive’s perspective. This is the reason for the development of OWL (web ontology language). Each of the important RDF Schema terms is either included directly in OWL or is superseded by new OWL terms.

In 2000, DARPA (Defense Advanced Research Projects Agency) started a project to achieve a machine-readable representation of the web, and as a result DARPA introduced DAML (DARPA Agent Markup Language). Almost at the same time, a group of European scientists was working on the OIL (Ontology Inference Layer or Ontology Interchange Language) [45] that was based on the frame-based language and description logic concepts. Later, both the projects modified their products to merge into one, and as a result DAML+OIL [46] was born. DAML+OIL syntax was similar to the present versions of RDF/XML and was able to define a set of facts in an ontology. The successor of DAML+OIL was introduced as OWL. OWL [47] is the W3C standard ontology
language for the web, which is designed to promote the semantic web. The concept of semantic web [32] states that information should be presented in such a way that it could be understandable for humans and could easily be process-able for machines. Therefore, OWL design is compatible with extensible Markup Language (XML) as well as with RDF and RDF(S) (see Figure 2.1). The ontology written in OWL is syntactically closer to XML; in addition, it satisfies the conditions of RDF. Due to these qualities, an OWL documents can easily be embedded in all kinds of applications that can consume XML or RDF-based documents. As stated in the previous section, it is more powerful than RDF or RDF(s) and allows reasoning engines to take it as input in order to infer new knowledge. Moreover, it allows the different rule engines to integrate with it, so the applications developed based on OWL ontology are supposed to be more intelligent and heterogeneous. OWL 2 [48] is an extension of OWL, which holds some extended features and rationales. In OWL 2, several new constructs were introduced along with a meta-modeling enriching features to support the extended entity annotations such as DisjointUnion, NegativeObjectPropertyAssertion, object level cardinality, IrreflexiveObjectProperty and N-array datatypes. In addition to the new construct, OWL2 provides three sublanguages OWL2 EL, OWL2 QL and OWL2 RL. Using OWL2 EL it is possible to build ontology driven applications where polynomial time reasoning complexity is required. OWL2 QL has been designed to facilitate users to easily access and query over stored data in databases. OWL 2 RL is a rule subset of OWL 2 and allows users to integrate rule engines for the purpose of reasoning [49].

2.1.5 Ontology Development Steps

Several compelling reasons have been documented in literature about ontology usage in information engineering domain. Ontology is used to share common application understanding explicitly among humans and software agents which enables the reuse of knowledge [50].
Ontology development is necessarily an iterative process and there is not a single way that could be followed as a fundamental guideline [51]. Therefore, the ontology development approaches vary based on its intended usage. Literature suggests competency questions that help to gather the domain knowledge and to determine the relationships among objects [52]. Examples of the competency questions are: What is the domain that the ontology will cover? For what we are going to use the ontology? For what types of questions the information in the ontology should provide answers? Who will use and maintain the ontology [50] [53]. In addition to competency questions that help to outline the domain knowledge, literature suggests a widely adopted technique which is presented in [50], where researchers described seven steps as listed below to develop an ontology.

1. Determine the domain and scope of the ontology.
2. Consider reusing existing ontologies.
3. Enumerate important terms in the ontology.
4. Define the classes and the class hierarchy.
5. Define the properties of classes—slots.
6. Define the facets of the slots.
7. Create instances

2.1.6 Linked Open Data

Data is considered a set of quantitative or qualitative values that can be measured, collected and reported. The Open Data concept advocates that data should be freely accessible to all potential users. The Open Data concept is a part of other “Open” movements such as open software, open hardware and open government. The notion of open government data [54] (OGD) has emerged as an effective communication medium between governments and citizens. In recent years, a number
of countries have made their government data available to the public, and we can find several open data portals from the US\(^9\), UK\(^{10}\) and Canadian\(^{11}\) governments [55]. Open government data can be reused to develop policies for needs assessments in public sectors [56]. Figure 2.4 below explains the concept of open data, open government and big data graphically. Linked open data (LOD) [20] is defined as a set of best practices for the publishing and connecting of open-structured data on the web that was not previously linked. These may be as diverse as databases maintained by two organizations in different geographical locations, or simply a heterogeneous system within one organization that has not easily interoperated at the data level. LOD is based on the standard web technologies such as HTTP, and URIs (Universal Resource Identifiers) [57].

\[\text{Figure 2.2: Venn Diagram of the Open Data Concept}\]


Instead of using web technologies to serve web pages for human readers, LOD extends the web technologies by incorporating a resource description, model (RDF), such that the data become

---

\(^9\) www.data.gov

\(^{10}\) www.data.gov.uk

\(^{11}\) www.data.gc.ca
machine processable and can be consumed automatically by computers. This also facilitates open data from different sources to be connected in order to build a global data space referred to as a Web of data (WoD) [58]. Heterogeneous information is retrieved from the linked open data cloud (LOD cloud) and reused to develop linked data-based applications. Figure 2.4 displays the linked open data cloud. To integrate a new dataset to the existing datasets on the linked data cloud, several data publishing recommendations have been introduced [59]. For instance, the utilization of well-known URIs to represent the entities’ names, following a hypertext transfer protocol, adding additional metadata with URIs. Linked data incorporates RDF for data representation and the SPARQL query language is used to query the datasets [20].

![Figure 2.3: Venn diagram Describing Linked Open Data Principle](Source: Concept derived from [16, 17])

Figure 2.3 illustrates the difference between linked data, linked open data and RDF. A linked data set is basically an RDFized version of any relational and non-relational dataset. RDF employs a specific URI scheme that does not allow every RDF dataset to be merged with any other RDF datasets. Therefore, a linked data set can be defined as private for certain institutions such as banks and hospitals. An important step in RDF translation is the selection of well-known semantic
vocabularies that define the level of interoperability and granularity of a dataset. Well-known vocabularies facilitate a dataset to be linked automatically to the linked opened data cloud without further conversion.

Figure 2.4: Graphical View of Linked Open Data (LOD) Cloud  
(Enlarged image can be viewed at  http://lod-cloud.net/)

Due to semantic web benefits, several life science datasets have been transformed into linked data examples include LODD (Linked Open Drug Data) [60], ChEMBL [61], Bio2RDF [62], linked mutation data [63] and EMBL-EBI RDF services [64]. Furthermore, the linked data approach has also been adopted in the healthcare domain to semantically organize patient records [65] and to manage the cancer related health service. To query and visualize the biomedical data, in [66]
Researchers present a tool that adopts linked data practices to integrate the diverse cancer genome datasets.

2.2 Bio-Imaging Ontologies

Biomedical images published within the scientific literature play a central role in reporting and facilitating life science discoveries. This section provides a brief review of popular biomedical image ontologies and standards that have been developed so far for the annotation of biomedical images.

2.2.1 DICOM Standard and Ontology

DICOM, which stands for Digital Imaging and Communication in Medicine [67], is a globally recognized standard for the exchange of medical imaging information. Although it has been adopted widely to annotate medical imaging information, a few parts of DICOM do not provide consistent semantic terminologies. Therefore, a piece of information can be annotated in multiple ways with the non-semantic version of DICOM [68]. In recent years, numerous efforts have been made to transform the DICOM standard into an ontology to promote its widespread usage in medical domain. The transformation of DICOM standard into XML encoding\(^{12}\) was the first attempt towards the ontological version of DICOM standard [69]. OWL encodings with the names “DICOM Controlled Terminology”\(^{13}\) and Semantic DICOM ontology (SEDI)\(^{14}\) were then introduced. The DICOM controlled terminology comprises of 3116 classes and 18 properties, while the SEDI holds 1424 classes and 4606 properties.

\(^{12}\) http://medical.nema.org/medical/dicom/current/source/docbook/

\(^{13}\) http://bioportal.bioontology.org/ontologies/DCM?p=summary

\(^{14}\) http://bioportal.bioontology.org/ontologies/SEDI
2.2.2 The RadLex Ontology

With the growing trend of digitizing medical records, there is a need for a standardized lexicon to store and retrieve the digital records. RadLex[15][70] is a comprehensive lexicon of vocabularies to annotate radiological images. The RadLex ontology [71] has around 68000 terminologies that mostly accomplishes the needs of radiologists, software developers and system vendors. RSNA (Radiological Society of North America)[16] has developed and provided the terminologies that were used in the RadLex ontology[17] development. Another important part of RadLex Ontology is that, RadLex terms are mapped with DICOM [69] and SNOMED-CT to meet the requirements of radiologists. It provides the semantic vocabulary to encode the radiologic procedures and to retrieve the contents within radiology reports [72]. The RadLex Playbook is a distinct feature in RadLex; it was developed by integrating clinical indications, body parts and imaging techniques. Hence, it delivers a standard system for radiology procedure naming. The RadLex ontology on bioportal provides 46059 classes, 46553 individuals and 95 properties in version 3.12.

2.2.3 The Open Microscopy Environment

The open microscopy environment (OME) [8] provides the data model and ontology for the manipulation and storage of biological microscopic data. The OME data-model defines image data with five dimensions (e.g. X, Y, Z, Time, channel), region of interests and path(s) of light. The path of light is recruited to produce the optical microscopic images. With the OME data-model, multiple images can be grouped into different datasets [73] and these datasets can further be organized to form a project. Moreover, an image can be a part of multiple datasets, and these datasets can also

---

15 www.radlex.org  
16 http://www.rsna.org/  
17 http://bioportal.bioontology.org/ontologies/RADLEX
belong to several projects. The major drawback of the OME is that the OME ontology is not yet formatted in the OWL standard. Moreover, the available ontology lacks an upper level ontology; therefore, it does not fully support the semantic data integration over heterogeneous dataset [74].

2.2.4 Quantitative Imaging Biomarker Ontology (QIBO)

A biomarker is a measurable indicator of the severity or presence of some disease state. The quantitative imaging biomarker alliance (QIBA) is responsible for upgrading a quantitative imaging in clinical care such that imaging can support biomarker use in clinical trials. QIBA has introduced several imaging biomarkers that are used in the prediction of therapeutic outcomes. However, there was no formal modeling available until 2013 to render the biomarker-related data interoperable. In order to provide the standardized vocabulary to the quantitative imaging biomarker domain, Buckler et. al [75] introduced QIBO (Quantitative Imaging Biomarker Ontology). There are 619 semantic terms in the current version available on Bioportal. QIBO does not employ any upper level ontology. Therefore, cross-domain data integration is not feasible with the current version.

2.2.5 Radiology Gamtus Ontology

The Radiology Gamtus Ontology (RGO) [76] has been developed to provide a formal knowledge modeling to the differential diagnoses processes in radiology. In diagnostic radiology, the word Gamtus refers to the “lists of differential diagnoses of imaging observations”. The Radiology Gamtus Ontology (RGO) incorporates 1674 differential diagnoses, 19,017 terms, and 52,976 links between these terms. RGO is currently being used for automated generation of medical questions and analysis of clinical radiology reports [77]. To facilitate the development of other information,
the gamtus portal (www.gamuts.net) provides a web services that allows the exploration and reuse of the RGO contents.

2.2.6 The OntoNeuroLog Ontology

The OntoNeuroLog (ONL) Ontology is a multi-layered and multi-component formal ontology. It was developed to facilitate the NeuroLOG [78] partners by providing a formal semantic vocabulary to annotate the neuro images. The ONL ontology has been designed by adopting the DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) as a foundational ontology. Besides, the taxonomy of ONL ontology comprises of seventy terms about PET (positron emission tomography), segmentation, parameter-quantification dataset and T1-weighted-MR-dataset. Every tissue in the human body has its own T1 and T2 value. This term T1-weighted is used to indicate an image where most of the contrast between tissues is due to differences in the T1 value. The ONL ontology also provides semantic vocabularies for the image acquisition and processing phases. Therefore, the taxonomy of acquisition and processing module includes nine terms about to registration and segmentations. The current version of ONL ontology was used to integrate the heterogeneous legacy neuroimaging datasets in Neurolog project. Furthermore, the ONL ontology was employed in the development of OntoVIP ontology [79, 80], which provides the annotation vocabulary for neuroimaging to promote the usage and sharing of neuroimaging process simulations.
2.2.7 The BIM Ontology

Existing ontologies and vocabularies, describing biomedical images (particularly sequence images), do not provide all the necessary semantic vocabularies for image annotations generated automatically and/or semi-automatically in SEBI project [15]. Therefore, we present an open ontology for the annotation of biomedical images (BIM) [23] scripted in OWL/RDF. The BIM ontology provides semantic vocabularies to describe the manually curated image annotations as well as annotations generated by online bioinformatics tools/services using content extracted from the sequence images by the Semantic Enrichment of Biomedical Images (SEBI) system. The BIM ontology is represented in three parts: (i) image vocabularies - which hold vocabularies for the annotation of a biological sequence image and/or region of interests (ROI) inside an image, as well as vocabularies to represent the pre- and post- processing status of an image; (ii) text entities – which cover annotations from the text that are associated with an image (e.g. Image captions) and provide semantic representation for NLP algorithm outputs; and (iii) a provenance model - which contribute towards the maintenance of annotation versioning.

2.3 Techniques for Curation and Manipulation of Annotations

This section introduces the widely adopted annotation curation and annotation manipulation techniques by employing the semantic web technologies. Semantic web techniques such as semantic web services, linked data and ontology data modeling make the curated annotation interoperable and reusable.
2.3.1 Techniques for Authoring Annotations

The use of semantic annotations within search algorithms has transformed the overall information discovery process [81] [82] [83]. However, a widespread availability of semantic annotations is required for new and existing data residing on the Internet to facilitate the full-scale benefits of semantic search. Primarily, the annotation generation process can be divided into three categories, namely: automatic, semi-automatic and manual. Manual annotation is one of the most accurate modes to generate annotation. However, it is a time-consuming and laborious process. Moreover, the manual annotation generation process requires domain experts to do the annotation. Nonetheless, with the current proliferation of data on the Internet, it is not feasible to annotate all the web data manually. Therefore, semi-automatic or automatic procedures are preferred. During the semi-automatic annotation procedure, intelligent algorithms are recruited to produce annotations while a user verifies the correctness of generated annotations. The third form is the automatic way to produce annotation on data. Essentially, web services are used to generate annotations automatically while a user then verifies the validity of the annotations gold standards rules/databases are considered [84].

Moreover, the selection of the annotation curation process depends on the type of data and its usage. Primarily, automatic annotation methods are considered appropriate when data is available in text format such as the web pages and documents available on the Internet. Similarly, to annotate technical documents such as PubMed articles, semi-automatic annotation-generation algorithms are used [85] [86]. Image and video contents are vague and most of the automatic algorithms are unable to generate accurate annotations on the contents. Therefore, the manual annotation procedure is preferred to annotate multimedia data [87]. In a semantic annotation process [15, 88], instances are mapped to their appropriate ontology classes and data is linked through semantic
properties. Available semantic annotation platforms are mainly classified into pattern-based and machine learning based. Nevertheless, limited numbers of platforms have been proposed that claim to support a multi-strategy annotation scheme.

A multi-strategy annotation scheme is a hybrid approach that works by combining both machine learning and pattern-based schemes. The pattern-based approach holds a corpus or multiple corpora containing manually defined information extraction patterns. The pattern-matching algorithm follows a series of information extraction rules to discover the relevant patterns in the text. Examples of pattern based annotation platforms are: AeroDAML [85] and Armadill [89]. With the AeroDAML web-based application, a user enters a URI of the webpage and the AeroDAML toolkit processes the page automatically and returns the annotations that are normalized with a default generic ontology of commonly found word classes and relationships. Moreover, users can set any particular ontology and invoke third party NLP services as well in the AeroDAML environment. AeroText [90] is one the customized application which is based on the AeroDAML core infrastructure and helps to develop high performance information extraction system for NLP-based content analysis applications.

Armadillo [89] applies a statistical evidence based technique to generate annotation on a document or a web page. An example of Armadillo could be to find faculty names with management role. Armadillo takes the web URI as an input and first applies the named entity recognition technique to gather the names. Armadillo then loops through two repeating phases: 1. evidence building and validation and 2. extraction of potential knowledge that find the persons based on the statistical evidence.
Moreover, most techniques based on machine learning employ some kind of probabilistic or inductive reasoning such as MnM and Ont-O-Mat. Probabilistic semantic annotation platforms work based on statistical models that predict the relevant entity location within the document [88].

2.3.2 Use of Semantic Web for Manipulating Annotations

Over the last decade, a small community of researchers has adopted semantic technologies for data integration and manipulation. In the Semantic web [32] paradigm there exists a common framework comprising of standards and technologies that enable data to be shared and reused across applications. Specifically, the semantic web advocates the use of an explicit data model to describe data, in an unambiguous way, such that independently generated data sets can be easily integrated under the same data model. This type of data integration is critical in a number of domains. The current proliferation of semantic web methodologies results from the use of cutting-edge standards such as the resource description framework (RDF) [36], RDFS [36], OWL [47] and SPARQL (SPARQL Protocol and RDF Query Language) [91], which are supported by several applications, due to their robustness and efficiency.

Web services [92] are considered a useful medium to leverage software functionalities in distributed environments without deploying the entire application on a client machine. A plethora of Bioinformatics software and services are available on the Internet, but most of them have their unique access criteria and information exchange formats [93]. In order to create conventional web services such as WSDL and REST [94], a developer requires in-depth understanding of the application’s architecture since web services follow XML-based structures to generate their schemas. Additionally, due to diversity in schemas of available web services, it is almost impossible to integrate the outputs of desired web services without transforming their outputs programmatically into a consistent format this manual transformation of schema demands a deep
understanding of underlying service architectures. A user cannot get the full benefit of these utilities unless they are available in an integrated and interoperable format. Semantic web services are the combination of semantic web and conventional web services. Unlike XML-based web services [19], semantic web services provide semantic metadata describing their input and output. This enables automatic discovery, composition, interoperation, and ad-hoc consolidation of the outputs as long as they are modeled in term of the same or compatible ontology(s). SADI (Semantic Automatic Discovery and Integration) [95] is a semantic web service. SADI framework is an example of best practices that allows the integration of and interoperability among resources on WWW by utilizing Semantic Web standards at all levels of the Web service stack. Instead of introducing new standards, SADI uses a set of HTTP-based recommendations that are well established and already in practice, such as RDF [S] and OWL for data representation and modeling. The SADI service consumes input RDF document(s) from the client, attaches new properties to the input URIs, and produces the output RDF documents. The semantic descriptions of the SADI services are represented by these properties, which are uniquely defined. These descriptions make the semantics of the underlying service functionality transparent to clients and facilitate automated service discovery and pipelining. SADI service can work standalone or in arrangements with other SADI services to get a consolidated RDF output.

2.4 Semantic Web Services and BioNLP

Web services [92] are considered to be an effective medium for the use of software functionalities in distributed environments without deploying the entire application on the client machine. A user cannot get the full benefit out of these utilities unless they are available in an integrated and interoperable format. Web services are platform independent and provide access to software that cannot be installed on desktop computers due to their complexity and processing requirements.
Over the last decade, biomedical natural language processing (BioNLP) has been validated, as a solution to address the text-mining and information extraction needs of life scientists. Recently the number of NLP and BioNLP tools published as web services have been growing steadily. There are several providers of text mining, web services, which include popular providers such as Whatizit [96], eLico [97], NaCTeM [98] and the Manchester Interdisciplinary Biocentre. Web services are typically registered in public catalogues (registries) e.g. BioCatalogue [93] and do not require installation. Most BioNLP tools produce XML based output, where XML schema represents the syntactic structure of the input and output messages. In many use-cases, integration of several text-mining, web services are required and the output results must be consolidated. Since XML schemas differ in their structure and do not provide the “meaning” of the syntactic XML elements, integration of web services and consolidation of results cannot be automated and requires additional programming work. Achieving the semantic interoperability - where distinct data not only share the syntax (same structure) but also the same semantics (elements linked to the same vocabulary which make possible system-independent interpretation of the data) is a bottleneck of XML-based approaches. Unlike XML-based web services, semantic web services provide semantic metadata describing their input and output. This enables automatic discovery, composition, interoperation, and ad-hoc consolidation of the outputs as long as they are modeled in terms of the same or compatible ontology(s). Despite the existence of several semantic web service frameworks such as OWL-S [99], SSWAP [100], WSMO [101], few of these have been adopted.
2.4.1 The SADI Framework

In our thesis, we leverage the Semantic Automated Discovery and Integration (SADI) [95] framework. The choice is based on the superior functionality of the framework for developing, and deploying of semantic web services, the available plug-in tools to and the availability of client software that simplifies the discovery and utilization of the services by end users [102]. Moreover, SADI has been successfully deployed to achieve semantic interoperability between data retrieval and data processing resources in several domains such as personalized medicine, clinical intelligence [103], Ecotoxicology [104], lipidomics [105], mutation text-mining [106].

SADI [95] is a set of best practices that are used to design stateless semantic web services; it is the byproduct of conventional web services and semantic web technologies that consumes and produces the RDF graph. Primarily, the SADI framework is designed to achieve semantic interoperability among web services designed for different purposes. The working mechanism of SADI is very straightforward, during service modeling; the base URI remains same for both input and output classes. After service execution, new properties appear with RDF output nodes. At the service development level, a developer needs to define the input and output classes in RDF/OWL format. In SADI framework, input and out class are defined in OWL and called service interface documents.

Both input and output classes has properties which can be restricted by defining the data types and cardinalities such rdf: type and xsd: string. SADI handles data interoperability without any syntactic transformation because of its service description format. Figure 2.5 shows the transaction steps for a Phmmer SADI service, which retrieves the species name, kingdom and sequence target
information when a sequence ASCII is provided. A series of transactions occurred between the client (on left) and the service endpoint (on right) using GET and POST.

Figure 2.5: A SADI Service Execution Steps
The root node sequence:1 attaches species name, kingdom and sequence target using output property hasKindom, has Species and hasTarget. The table 2.1 below presents a comparison between SADI and other web service registries. The SADI framework algorithm generates the skeleton for semantic service while a developer is required to write the business logic in a stub section of an auto-generated file.

Figure 2.6: BioNLP-SADI in Context

Figure 2.6 displays the position of BioNLP SADI among available techniques. The three circles in Figure 2.6 correspond to the web services, semantic web and BioNLP respectively, whereas the SADI, Whatizit and BioNLP SADI are the offshoots of these technologies. SADI provides transportation and presentation layers’ tasks without human intervention. The available plug-in tools and the availability of client software, simplify the discovery and utilization of the services by end uses. Semantic web and related technologies provide a broad range of available RDF/OWL tools - RDF/OWL APIs, SPARQL, semantic databases (triple stores), semantic reasoners. Therefore, they can facilitate out-of-the-box use of system results in the different use case
scenarios such as search and data analysis, integration with other data, browsing and representing data, data integrity evaluating, text-mining and analysis etc.

Table 2.1: Feature Comparison of SADI with Frameworks

<table>
<thead>
<tr>
<th>Framework Name</th>
<th>Framework Type</th>
<th>Internal Working</th>
<th>Service Composition</th>
<th>Data Extraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>SADI</td>
<td>Semantic</td>
<td>RDF</td>
<td>Yes</td>
<td>SPARQL</td>
</tr>
<tr>
<td>Feta</td>
<td>Non-Semantic</td>
<td>Input, output, operation type, Task</td>
<td>N/A</td>
<td>Ontology based keyword</td>
</tr>
<tr>
<td>Magallanes</td>
<td>Non-Semantic</td>
<td>String matching</td>
<td>Yes</td>
<td>Keyword</td>
</tr>
<tr>
<td>EMBRACE</td>
<td>Syntactically annotated</td>
<td>String matching</td>
<td>N/A</td>
<td>Keyword</td>
</tr>
<tr>
<td>BioMoby</td>
<td>Resource/Object Type</td>
<td>Resource type, I/O</td>
<td>N/A</td>
<td>Keyword</td>
</tr>
<tr>
<td>myExperiment</td>
<td>Tags</td>
<td>String matching</td>
<td>N/A</td>
<td>Keyword</td>
</tr>
<tr>
<td>Taverna</td>
<td>BioMoby metadata</td>
<td>String matching</td>
<td>Workflow</td>
<td>Keyword</td>
</tr>
<tr>
<td>BioCatalogue</td>
<td>Categories, some tags</td>
<td>String matching, Categories, filters</td>
<td>N/A</td>
<td>Keyword</td>
</tr>
<tr>
<td>SSWAP</td>
<td>Third-party ontologies</td>
<td>RDF</td>
<td>N/A</td>
<td>Query Graph</td>
</tr>
</tbody>
</table>
2.4.2 SADI Web Service Modeling

SADI sequence analysis and BioNLP web services have been designed by exposing the sequence analysis and BioNLP software’s as web services to annotate the biomedical sequence images. BioNLP web services annotate a biological sequence image by extracting the information from the image associated text (e.g. caption, description). To demonstrate the working of SADI, this section explains the SADI modeling process by introducing the use cases of a sequence analysis and a BioNLP web service [12]. Figure 2.7 displays a Phmmer [113] web service modeling. The Phmmer SADI web service takes a protein sequence as input and searches against protein sequence analysis resources to retrieve all possible HITS (High Identity with Tolerance). The input class in Phmmer service has a hasProteinSequence property which accepts the a sequence input of xsd:string type. On service execution the Phmmer_Output class generate new instances and attaches that to hasSpecies, hasKingdom, hasTarget and hasBitScore properties. The SADI sequence analysis service modules yield new annotations to a sequence image such as the protein family, the species or motifs [114].

Figure 2.8 displays a modeling Drug extraction SADI web service. The getDrugFromText_Input class has a hasContent property that get a text string as an input and invokes the service. The getDrugFromText_Output class generates string type annotations defining the drug mentioned in the text along with its position in the test. The pink blocks in Figure 2.8 represents the semantic modeling of getDrugFromText output modeling. The goal of drug extraction service is to detect the drug mentioned in the image caption and the related description and extract them. The service is represented by modeling the relation between the document and annotations of type Sentence in text segments. The input and output classes as displayed in Figure 2.7 and Figure 2.8 are called service interface documents and are defined in terms of OWL-DL classes [115]. These documents
consist of property restrictions on these classes by referring to specific data-types required by the service. SADI services consume an RDF instance, as input and produce an RDF instance, as output, both of which are compliant with the definitions of input and output classes, respectively.

Figure 2.7: A Graphical View of the Phmmer SADI Service Modeling

Because of the uniform representation of input and output documents as RDF, data interoperability is achieved without the need for any kind of syntactic transformation, which is characterized as one of the most important advantages of adopting SADI [95]. The classes in the interface documents exist on the Web in OWL ontology document(s) and the URIs of these classes must be resolved to that ontology by the service endpoint using HTTP GET. SADI input and output instances could be a large RDF graph. For example, a document may have hundreds of drugs mentioned in it which would appear with the output property of the service. SADI conventions dictate that a URI will represent a root node, around which both the input and output instance must be classified with rdf: type property. Thus, the URI of the output instance is the same as the URI of the input instance.
Since the sequence analysis SADI services work as a standalone component, users can utilize these services according to their needs in multiple application development scenarios.

Figure 2.8: SADI Service Modeling for a Drug Extraction Service

2.4.3 Consolidation RDF Outputs from SADI web Services

Using SADI as a platform guarantees results of text mining and sequence analysis are interoperable with SADI services which access on structured data. This can take the form of adding new data, verifying text extracted results against existing data in knowledge bases or controlled vocabularies, a process known as grounding (assuming that knowledge bases have a compatible semantic interface). Figure 2.9 shows a (prototypical) RDF graph automatically assembled from the merged output of the three services. Document_1 was annotated with drugs by the Drug-Extraction-Service and was split into sentences by the Sentence-Splitter-Service. Following this is the Drug-
Drug-Interaction-Service, which finds all drug pairs with potentially harmful interactions. The Drug-Drug-Interaction service in our example is a data-retrieval service (not a text mining service).

![Diagram of Drug-Extraction and Drug-Drug-Interaction SADI Services Output Modeling](image)

Figure 2.9: Drug-Extraction and Drug-Drug-Interaction SADI Services Output Modeling

This service is based on the DrugBank [7] database that contains information about drugs and their interactions. The service retrieves, for each drug, all the known interactions from the database and attaches them to the drug as instances of the Drug-Drug-Interaction class (see Figure 2.9). The resulting output of three services makes it possible to pose queries for the target information such as, find all sentences (e.g. Avoid taking St. John’s Wort and Indinavir sulfate (Crixivan). where potentially harmful drug-drug interactions are mentioned.

### 2.4.4 Federated Querying

Due to the growing numbers of data sources, integration and manipulation of data have appeared as a challenging issue for life scientists. A number of solutions have been proposed so far to integrate data by adopting the centralized data integration approaches such as UniProt [116], KEGG [117], Bio2RDF [62]. The centralized integration approach demands continuous updates,
which is a cumbersome, and error prone activity. Conventional distributed data integration techniques depend heavily upon the existence of rigid schema that impedes the widespread data search. SHARE [25] has been developed to address integration and manipulation challenges of heterogeneous data resources. SHARE is an experimental federated query client that performs automatic discovery and orchestration of SADI services.

In a typical scenario, a user first looks up predicates required to build up a query, in the list of predicates declared as provided by SADI services in a registry, and also related classes and property predicates in the referenced ontologies. Available concepts are used to form a regular SPARQL query, and it is sent to a SHARE endpoint. Importantly, the SHARE engine discovers, in a registry of services, which services have to be invoked and in what order, to execute the query. Note that this qualifies for automatic discovery, composition and invocation [95]. The user needs only to understand the semantics of the URIs being used in the query, although knowing the services providing the predicates can be beneficial. In the next phase, SHARE invokes the SADI web service and stores the results in a virtual triple store. SHARE further filters the virtual triple store entries to fetch the desired results. SHARE was designed as an experimental tool therefore it could not efficiently handle complex queries [25]. Hydra\textsuperscript{18} [103], a new commercial federated query engine for SADI services, notably resolves the heterogeneous data discovery and ad hoc integration challenges present in SHARE.

### 2.5 Biomedical Images

Biomedical Imaging refers to techniques that capture and process biomedical images in order to derive quantitative information and further utilize it for diagnostic and therapeutic purposes [118]. Biomedical Imaging itself is a multidisciplinary field which incorporates mathematical algorithms,

\textsuperscript{18} http://ipsnp.com/hydra/
engineering techniques, chemical operations and image capturing methodologies to acquire the biomedical images ranging from molecular/cellular images to mouse images to large animal/human images. Biomedical imaging employs various techniques such as X-Rays (CT scans) [119], sound (ultrasound), magnetism (MRI), radioactive pharmaceuticals (nuclear medicine: SPECT, PET) or light (endoscopy, OCT) [120] to evaluate the status of an organ or tissue and to monitor a patient over time to establish diagnoses and treatment prognoses. This section provides a brief overview of current biomedical imaging modalities.

2.5.1 Radiography

Radiography is one of the most prominent biomedical imaging techniques that have been used for diagnostic purposes. To obtain a radiographic image electromagnetic radiation, particularly in the form of X-rays, is passed through a dense and non-uniform object such as the human body. Heterogeneous X-rays are produced with the help of an X-ray generator while a photographic film is used on the other side of the object to obtain the substance’s interior image. Primarily, radiography is classified into two categories namely: projection radiography and fluoroscopy radiography. The projection radiograph (X-rays) process is less expensive than the fluoroscopy and is mainly employed to examine the severity of fractures [119]. In special cases, the projection radiography is considered favorable for investigating the pathological changes in the lungs. The distinguishing characteristic of CT (computed tomography), which is a variant of X-rays, is that the radiation source moves while the subject remains stationary. Computers compose a 3D image from the x-ray absorption pattern.

However, in fluoroscopic radiography, different types of contrast media such as barium, iodine, and air are used to generate a movie style representation of a body organ. Usually a monitor is attached to the fluoroscopy machine that projects live output [120]. Fluoroscopy operates with low
potency X-rays to dynamically visualize the internal working of an organ. A fluoroscopy device has a beam translator that helps to translate the radiation data into a comprehensible image. Figure 2.10 displays a radiography image block of a human backbone that has been taken at different stages of a treatment.

![Radiography image block of a human backbone](image)

**Figure 2.10: An example of a Radiograph [121]**

### 2.5.2 Magnetic Resonance Imaging (MRI)

Magnetic resonance imaging (MRI) [88] is another technique where a scanner interprets magnetic field or radio wave pulses to generate comprehensive photos of body organs and tissues situated inside an organism. MRI is the most widely used diagnostic procedure after X-rays in the medical domain. Most MRI machines resemble a tube with a mounted magnet on the upper frame of the machine. Since MRI applies magnetic resonance instead of any radioactive substance it is considered to be a safer way to produce medical images. MRI is more sensitive than computed tomography and produces the high quality visualizations therefore it is particularly useful as an investigative tool in neurological cancers [4].
Moreover, Functional MRI (FMRI), which is an extension of MRI, helps to measure brain activity by detecting associated changes in blood flow [120, 123]. Figure 2.11 demonstrates the functional MRI of a brain.

2.5.3 Fluorescence Microscopy

Fluorescence microscopy [124] functions in a similar way to conventional microscopy, with the sole difference being that it uses a fluorescent substance to generate the magnified images. Conventional microscopy relies on visible light to illuminate and capture a magnified image. However, in fluorescence microscopy, a relatively high-pitched light source that can stimulate a fluorescent species in a sample of interest is used [124]. This fluorescent species in return emanates a small energy light of a longer wavelength that yields the magnified image, instead of the original light source. Figure 2.12 displays fluorescent microscopy inside a cell. The technique of
fluorescence microscopy is mainly applied to observe the structural components of cells to measure the cell growth and to capture the images of genetic material within a cell.

Figure 2.12: A view of Fluorescent Microscopy Inside a Cell [125]

2.5.4 Histogram Images

In histology, the microscopic anatomy of cells and tissues is studied by taking a tissue slice. A regular light microscope is used to collect the samples in general experiments, whereas in certain cases electron microscopy is employed. Unlike fluorescent microscopy, where fluorescent material has been used to illuminate the area of interest, in histology a standard color camera system is employed [126]. Figure 2.13 displays a histogram image of a cell.
2.5.5 Gel Electrophoresis Images

In gel electrophoresis [128], macromolecules such as protein, DNA and RNA are separated and examined. The analysis is mainly conducted based on a macromolecule’s physical traits such as their size and the charge that they carry. An electric field over a gel results in the DNA moving through the gel matrix at an inverse rate to fragment size.
The speed and distance covered by the molecules depend on their size. Generally, the molecules that are short in length cover more distance, compared to a DNA molecule with a larger size [128]. The main usage of gel electrophoresis process is to find the genes responsible or associated with diseases.

2.5.6 Biological Sequence Images

A biological sequence is a single, continuous molecule of nucleic acid or protein [130] which can be represented as DNA (deoxyribonucleic acid), RNA (ribonucleic acid) or protein sequences. The DNA blueprint consists of four alphabet of chemical compounds called adenine (abbreviated A), cytosine (C), guanine (G) and thymine (T) which attached themselves to the sugar/phosphate to formulate the complete nucleotide. The DNA molecules act as a storage to hold the genetic information of organisms [131]. DNA and RNA share the similar chemical structures with the difference being the binding sugar and the nucleobase. The RNA has the sugar ribose in its structure while the DNA holds sugar de-oxyribose in its structure. Moreover, RNA has the nucleobase uracil while DNA contains thymine [132].

Similarly, a protein sequence holds twenty amino acids in its structure which are: Alanine (A), Arginine (R), Asparagine (N), Aspartic acid (D), Asparagine (B), Cysteine (C), Glutamine (Q), Glutamic acid (E), Glutamine (Z), Glycine (G), Histidine (H), Isoleucine (I), Leucine (L), Lysine (K), Methionine (M), Phenylalanine (F), Proline (P), Serine (S), Threonine (T), Tryptophan (W), Tyrosine (Y), Valine (V). All biological sequences have more common features than differences. As it is displayed in Figure 2.15, these amino acids can come in any order to build a sequence string. In the biomedical literature, sequences are usually presented in figure format. Since a biological sequence holds important information, which can be helpful to understand an
organism or its cellular features, function, structure and evolution, biological sequence analysis has gained importance in biological experiments [133].

Figure 2.15: An Example of a Protein Sequence Image found in Biomedical Literature [134]

Sequence analyses are performed through various sequence alignment techniques. In sequence alignment protein and DNA sequences are arranged to find optimal regions of similarity. Global and local alignments are two popular techniques to find the sequence alignments. The global alignment technique seeks to align every residue in each sequence and largely is applied when the sequences are of equal length [130]. An example of a global sequence alignment algorithms is the Needleman–Wunsch algorithm [135]. To find the similar motifs in the large sequence strings, the local sequence alignment technique is recommended. BLAST [86] is popular tool used to search local alignment among sequences. Researchers can identify gene function by studying the similarity between the compared sequences. To infer the homology and to detect the evolutionary relationships among sequences of protein and DNA multiple sequence analysis [136] techniques are used. Usually more than two sequences of similar or different in length are used as input in multiple sequence analysis algorithm to infer homology. Figure 2.16 displays a sample literature image containing multiple sequences.


2.5.7 Graphs and Charts

Graphs and charts are a common way to illustrate a set of data diagrammatically. In biomedical publications, graphs and charts are often used to define change in a system. The change could be positive, negative or neutral. Moreover, graphs and charts are an easier way to define the comparison between the functionality of two systems. Bar graph and pie charts are most widely used graph types [120].

2.6 Sequence Analysis Tools and Databases

This section provides an overview of the sequence analysis tools that have been employed in SEBI.

2.6.1 Basic Local Alignment Search Tool (BLAST)

BLAST [138] is one of the most common sequence-searching programs used in Bioinformatics. It has been developed to compare a protein or a DNA sequence against the respective protein and DNA databases (e.g. UniProt, protein sequence). The specific goal of BLAST is the homology detection between strings of biological sequences. BLAST employs a heuristic search method and compares strings of protein or DNA sequences. The algorithm looks for short potential matches
between the two sequences instead of comparing complete full-length sequences. The process of
selection a short match between the sequences is called seeding, where the string of interest i.e the
query plays an important role while finding the similarity between sequences. BLAST provides
four different algorithms designed to accomplish the specific tasks. BLASTp compares a protein
sequence query to protein databases. PSI-BLAST extends this and allows users to build a PSSM
(position-specific scoring matrix) using the results of the first BLASTp run. Similarly, PHI-BLAST
performs an alignment search and limits the alignments to those that match a pattern in the query.
In specific cases, prior to searching for matches to a DNA or protein sequences, the BLAST
algorithm performs a six-frame conceptual level translation.
DELTA-BLAST however, constructs a PSSM using the results of a conserved domain databases
(CDD) search and looks for a sequence database. Consider the amino acid sequence that holds
‘GLKFA’ a short sequence of interest, after executing with BLASTp to find the similarity. At the
start, the BLASTp algorithm constructs sets of three letters from the query string such as: GLK,
LFK and KFA and seeks for the best match of these three newly generated sub-strings against
protein databases. Next, BLAST calculates an alignment score before selecting a region of interest
between the sequences and sequences in the target databases. BLAST uses a scoring matrix called
BLOSUM62 to calculate the alignment. BLOSUM62 is considered as the optimal sequence-
matching scoring technique [139]. BLAST algorithms can be used for finding various interesting
annotations: for instance, analyzing a sequence with BLAST can identify the species type. This is
helpful when the scientists deal with unknown genes of same function. A phylogenetic tree can
also be generated from the BLAST outputs, which supports phylogenetic analysis which could be
used to locate the position of a gene on chromosome of a particular species.
2.6.2 HMMER

HMMER (Hidden Markov Model-based sequence alignment tool) [113] was developed to provide easy access to sophisticated sequence homology detection algorithms. HMMER’s functionalities have been incorporated in core modules of Pfam and Interpret. With the release of HMMER 3.0 in 2010, the current version of HMMER is capable of delivering around 100-fold sensitivity improvements, compared with its previous versions. The new HMMER 3.0 version is regarded as more efficient than the BLAST (see section 2.6.1), a well-known tool to analyze a protein sequence. HMMER utilizes a profile-HMM that is a statistical model. The Profile-HMM (Hidden Markov Model) defines a system as a Markov process with number of hidden states. The HMMER software package facilitates homologous protein and nucleotide detection through matching the profile-HMM with sequence databases.

Unlike BLAST, HMMER adopts the log-odds likelihood scores summed over alignment uncertainty scores or simply a forward scoring scheme. The forward scoring scheme is considered to be a better detector of distant homologs as there are several possible ways to align a distantly related sequence. Older versions of HMMER were client side installations therefore a user had to install the entire software on a UNIX machine and the only option to access the system was through the command line. The enhanced algorithm in HMMER 3.0 makes it possible to access the core functionality of software through custom designed web services. At present it is feasible for a user to invoke the HMMER algorithm as a component in the development of bioinformatics applications. The official HMMER portal comes with three different versions of HMMER.

Each HMMER version has been modified to carry out a specific task [97]. The Phmmer (version 3) is similar to BLASTP in that it takes a FASTA formatted sequence as input and compares against potential protein sequence databases. FASTA format is a text-based format for
representing either nucleotide sequences or peptide sequences, in which base pairs or amino acids are represented using single-letter codes. Phmmer translates an input sequence into a profile HMM (Hidden Markov Model) and searches for protein sequence databases to identify the possible alignments. In contrast, the Hmmscan tool takes a sequence string input similar to Phmmer, but searches for possible alignments in the Pfam[140] profile HMM library. The outcome of a Hmmscan provides a bit score and an E-value showing the degree of relatedness to known Pfam protein families. The third tool on the HMMER portal named as ‘Hmmsearch’ has been designed to search profile-HMM against protein sequence databases. The input function of Hmmsearch accepts input in multiple formats such as: Clustal[141], MSF, SELEX, STOCKHOLM and aligned FASTA[142].

2.6.3 Protein Family (Pfam)

Pfam is another protein database that contains protein family information along with multiple sequence alignments that have been translated into hidden Markov model profiles. A Pfam search allows user to submit a sequence and discover the related protein families[143]. The process to prepare and search protein family profiles can be divided into four phases as below.

At the first stage, an algorithm like clustal x[100] is used to generate a high-quality sequence alignment known as a seed alignment.

The second stage involves the construction of a hidden Markov model from the seed model for further processing. In the third stage, the HMMER algorithm searches a library of a profile HMMs (Pfam) for a match to the query sequence. The last stage, using a family-specific sequence and domain gathering threshold (GAs)[144] searches for full-length alignment to a protein family HMM.
2.6.4 PROSITE

PROSITE [145] is well-known protein database that contains the information about protein families, their domains, functional sites and conserved amino acid patterns. PROSITE has been manually curated and coupled with Swiss-Prot. The major usage of PROSITE is to determine the functionalities of a newly discovered protein and run an analysis of undetermined activities of known proteins. The properties of well-studied genes can be propagated to biologically related organisms, and in cases of unknown genes their biochemical functions can be predicted based on their similarities to sequence of known genes. The ExPASy proteomics analysis [146] servers of PROSITE provide a number of sequence analysis and motif detection features. PROSITE also incorporates ProRules that offer supplementary information about functionally or structurally critical amino acids. The supplementary information can be grouped into active sites, substrate or cofactor-binding sites, posttranslational modification sites or disulfide bonds, to help determine protein function [147].

2.6.5 Conserved Domain Database (CDD)

The Conserved Domain Database (CDD) [148] is a collection of manifold sequence alignment and supplementary models that have been imported from a number of biological resources such as Pfam, SMART, COG, PRK and TIGRFAM. These manually curated models provide detail about 3D-structure information used to explicitly define the domain boundaries, align blocks, amend alignment details, and provide insights into sequence, structure and function relationships. CDD models have been organized in a hierarchical manner that makes the information non-redundant and further aids the process of looking for protein domain families. Moreover, the conserved domain databases hold information about domain footprints and functional sites through the Entrez query and information retrieval system [120].
2.7 An overview of Fuzzy Logic

Fuzzy logic [149] is a form of many-valued logic that deals with reasoning that is approximate rather than fixed and exact. In classical set theory, elements either belong to a particular set or not. The concept of partial membership does not exist in classical set theory. However, in fuzzy set theory the association of an element with a particular set lies between 0 and 1 and is called its degree of association or membership. In our daily life, we find many vague statements such as: hot water, cold weather, dark night, high danger, etc. We cannot quantify exactly the severity of the danger or hotness. The fuzzy set theory adds a generalization concept to classical set theory and makes it diverse enough to represent imprecise boundaries like hot, tall, low speed, high risk, etc. Fuzzy set theory can be explained formally as:

**Definition 1.** A fuzzy set ‘s’ over the universe of discourse ‘X’ can be defined by its membership function $\mu_s$, which maps element ‘x’ to values between [0,1].

$$\mu_s (x) : X \rightarrow [0,1]$$  \hspace{1cm} (1)

In the above equation, $s$ is the fuzzy set and $\mu$ is the degree of membership. Here, $x \in X$ and $\mu_s (x)$ represents the degree of membership by which $x$ belongs to $X$. And $x$ is considered to be a full member of $X$ if $\mu_s (x) = 1$ and is considered to be a partial member if $\mu_s (x)$ is between 0 and 1, say 0.65. If $X$ is continuous, then $S$ can be written as follows:

$$\hat{S} = x \mu_s (x) / x$$  \hspace{1cm} (2)

A fuzzy set $\hat{S}$ over the universe of discourse $X$ can be organized into an ordered set of pairs:

$$\hat{S} = \{(x, \mu_s (x))|x \in X \}$$  \hspace{1cm} (3)

**Definition 2.** Let $X$ and $Y$ be the two universe discourses. A fuzzy relation $R(x, y)$ is a set consisting of the product space $X*Y$ in a membership function.

$$R (x,y)=\{(x,y),\mu_R(X,Y)|(X,Y)|\in X*Y\}$$  \hspace{1cm} (4)
In compliance with fuzzy set theory, $x$ and $y$ are considered to be fuzzy sets in the product space $X \times Y$. The variable $\mu$ is the fuzzy membership degree. A fuzzy relation represents the degree of presence or absence, interaction or interconnectedness between the elements of two crisp sets.

### 2.7.1 Fuzzy Membership Function

The membership function of a fuzzy set is a generalization of the indicator function in classical sets. In fuzzy logic, it represents the degree of truth as an extension of valuation (A valuation is an assignment of truth values to formal sentences that follows a truth schema.) [150]. Figure 2.17 explains the concept of fuzzy membership function graphically. In figure 2.17, the X-axis represents my monthly spending at UNB and the Y-axis shows the possible fuzzy membership values. In this particular case, we can translate $400 to its fuzzy membership value, which would be in the 0.2 and 0.5 range of fuzzy membership. Likewise, if I spend $780/per month then its corresponding membership value would be 0.3 and 0.5 for medium and high respectively.

![Figure 2.17: Fuzzy Membership Function](Image)
Fuzzy Inference Engine (FIE) is a key part of fuzzy based reasoning systems [150]. It uses fuzzy rules to devise the correct conclusion from the vague datasets. A fuzzy rule engine consists of four parts named: 1- fuzzification stage, 2- Rules knowledge base, 3- Inference chamber and 4- Defuzzification module. The general architecture of the fuzzy rule engine [150]. is shown in Figure 2.18.

![Fuzzy Rule Engine General Architecture](image)

**Figure 2.18: Fuzzy Rule Engine General Architecture**

At the fuzzification stage, the crisp input is gathered and fuzzified by using the fuzzy linguistic variables, fuzzy linguistics terms and a membership function. Subsequently, output is derived based upon rules (see section 4.2.4 for rules). At the last stage, output is translated back to a crisp value.
Chapter 3

Semantic Sequence Image Processing

SEBI utilizes the information extracted from sequence images as seed data to harvest new annotations from heterogeneous online biomedical resources. This chapter explains the contributions to sequence image acquisition, OCR extraction and evaluation stages. The chapter has been divided into introduction, methods, results and evaluation and discussion sections.

3.1 Introduction

The introduction section has been arranged into three parts to provide a comprehensive overview of sequence image acquisition, optical character detection and extraction, as well as the OCR tools and technologies.
3.1.1 Sequence Image Acquisition

SEBI acquires biomedical images from the YIF [13] which currently holds over 1.5 million publication images along with their associated metadata in its index. The data in the YIF originate from open source PubMed articles under license from the NLM (National Library of Medicine) as XML files. At the image data manipulation (IDM) tier of the SEBI framework (See Tier 1 of Figure 1.2) we have designed the image acquisition algorithm (IAA). The acquired image data from YIF were redundant and often with incomplete metadata. Therefore, to resolve data redundancy and to populate missing image metadata, IAA establishes connection with YIF and the PubMed API concurrently to crosscheck the image metadata. All the relevant metadata (e.g. title, authors, PubMed IDs, and year for each article) are extracted and stored in MySQL for a quick look up, and in a triple store as parallel storage. The image manipulation engine (IME) at Tier 2 fetches images from MySQL and invokes the optical character recognition (OCR) [151] algorithm. The OCR process detects and extracts the optical characters by applying the DIA (deep image analysis) and SISA (sequence image segregator algorithm) algorithms. DIA primarily recognizes and separates the sequence images from other image types (i.e. radiographs, gel images etc.) and stores them in separate directories. SISA further analyses the sequence images to segregate the DNA and protein sequence images. Section 3.2.6 explains the sequence image segregation process.

3.1.2 Optical Character Detection and Extraction

OCR algorithms are designed to detect and extract the optical characters automatically from a digital image. While functional, these algorithms require training to increase their accuracy and response time. The IME employs Tesseract [152] to extract the optical characters from the sequence images. Tesseract is the most accurate optical character recognition API available to date.
under the Apache license. We trained our OCR algorithm and tested it against sample datasets (See section 3.2.6). Moreover, the OCR algorithm performance has been optimized to identify the optical characters even from low resolution sequence images. The detail of both the Tesseract and Leptonica image processing libraries [153] have been described in the section 3.1.3. The main advantage of employing the Tesseract engine to underpin the OCR algorithms, is that it can be trained with new font-faces or families i.e. Times New Roman or Verdana, and with multiple image layout styles i.e. portrait, landscape and, multi-page. Moreover, Tesseract is an open source library therefore, a large number of third party’s libraries and GUI tools can be configured for the Tesseract engine, to expedite the training and testing of OCR processes. While extracting the sequence ASCII strings from the sequence images, it was observed that our dataset also contained some multiple sequence alignment images, as shown in Figure 3.1. The OCR engine was trained initially to extract an entire character’s block residing in an image and parse it only as a contiguous string.

To recognize the multiple sequence alignment images, we initially distinguish the multiple sequence images from the single sequence images dataset based upon their captions or other metadata information. For example, the keywords mentioned in the image captions are processed with NLP web services. The text associated with Figure 3.1 has a multiple-alignment mentioned in its caption. The NLP web services [12] have been designed to detect and extract images with potential keywords indicating the presence of multiple sequence alignments. The OCR algorithm has been optimized to manage multiple sequence alignments by extracting the sequence strings line by line from multiple sequence images. Subsequently, the output (extracted line string) is re-assembled with rest of sequences mentioned in an image. The main purpose behind this process is to get all possible annotations associated with each image separately.
3.1.3 Overview of OCR Tools and Technologies

As mentioned in the previous section, we employed the open source Tesseract engine [152] in our algorithm OCR extraction. Tesseract essentially recruits the Leptonica image-processing library to identify a text block inside an image. Leptonica is an open source image analysis and processing library that has been scripted in C++. The Leptonica library [153] contains a number of useful programming functions that support the analysis of an image at different stages of OCRs. The current version of the Tesseract API can process a large number of image formats and can detect characters in over 60 different languages. Tesseract was developed by the Hewlett Packard (HP) in 1985 as a proprietary software. Hewlett Packard (HP) and University of Nevada released the open source version of Tesseract in 2005. Since then several organizations have contributed to the Tesseract API to enhance its functionality. Google also played a key role in Tesseract development and have released several open source Tesseract versions for Windows, Linux and Mac OS X. To
extract the optical characters from the sequence images in the SEBI framework [15], the image manipulation engine (IME) at Tier 2 utilized TESS4J, a Java wrapper for Tesseract-OCR which includes a number of image filters such as hue, saturation and brightness filters. The Tesseract API is remarkably powerful and provides various customizable features. However, to increase the character level accuracy and response time, it does require domain specific training [151, 154].
3.2 Methods
The methods section provides detail about preprocessing of sequence images, OCR extraction and post-processing of sequence output. Moreover, it introduces our methods of sequence image classification. The global and local level classification procedures detect and extract the sequence images from the full image dataset and further classifies them into protein and DNA sequence images.

3.2.1 Sequence Image Preprocessing

The images in YIF repository are available in web friendly formats such as GIF and JPEG. Therefore, most of the images that we are acquired were of low resolution (200 × 100 pixels). However, to yield the maximum accuracy, an OCR requires images with at least 300*300 pixels resolution. To increase the resolution and to enhance the contrast and sharpness of sequence images, we employed image filters such as: gray scale, hue, saturation & brightness [155], Gaussian and Laplace Filters [154]. Figure 3.2 displays a sample of a sequence image downloaded from the Yale image finder before processing.

![Figure 3.2: Low Resolution Image from Yale Image Finder](image-url)
The sequence image in Figure 3.2 is blurred and with low overall resolution. *ImageJ* [156], an open source Java library was employed to carry out image processing. A Java program has been designed by utilizing the ImageJ library that could detect and reset the image resolution. The background of each images was removed and resolutions were enhanced to a minimum of 300×300 Pixels resolution before the application of the image filters. The image output after applying the gray scale filter can be viewed in Figure 3.3.

*Figure 3.3: An Image Post-resolution and Gray Scale Filter Adjustment*

A Gaussian blur filter uses the Gaussian mathematical function to smooth a gray scale image. It is widely used to reduce the image noise in many image-processing applications. In addition, discrete Laplace Filter [154] is applied for image edge detection and motion estimation. The overall objective of whole image processing is to obtain high resolution images that could help to generate OCR with maximum accuracy. The OCR algorithm generates inaccurate outputs with low resolution images. Both the Gaussian and Laplace filters have been applied on HSL treated images. HSL (hue-saturation-lightness) and HSV (hue-saturation-value) [154] were introduced in the
1970s to represent the color model of a point in a cylindrical fashion. The results after applying the Gaussian and Laplace filters can be observed in Figure 3.4 and Figure 3.5 respectively.

**CLUSTAL 2.0.2 multiple sequence alignment**

Figure 3.4: A Sequence Image after the Application of a Gaussian Filter

CLUSTAL 2.0.2 multiple sequence alignment

Figure 3.5: A Sequence Image View after Laplace Filter Application

The sequence image in Figure 3.5 has better resolution than the image in Figure 3.2. As a last step, a string detection histogram algorithm was applied that surrounds all characters with in an image (see Figure 3.6). This process shortens the overall optical character recognition processing time as
the optical character detection algorithm confines itself to search the text within the histograms and does not seek to process the whole image.

**Figure 3.6: A View of Sequence Image with Text Surrounded Histograms**

### 3.2.2 System Requirements for OCR Algorithm Training

To carry out training of the OCR engine, the Tesseract API version 3.0.2 was deployed on Ubuntu 12.04 along with an Intel Core i5 Processor and an 8 GB system memory (RAM). The jTessBoxEditor, which is a graphical box editor and trainer for Tesseract, was configured to accelerate the training process. The jTessBoxEditor works with JVM (Java virtual machine) 7 and can accept multipage images in TIFF format. To initiate the training process a set of 540 sample images were extracted from the sequence image repository based on their resolution and layout. The sample set was further divided into six groups with 90 images in each category. Images in each category were selected based on their resolution value such as: low (< 200×200), medium (⇒ 200×200 and ≤ 500×500) pixels, high resolution (> 500×500 pixels) and images with multipage (large size image) features. Tables 3.6 and 3.7 provide the sequence images signatures that
we used for OCR training and testing. Sequence images listed in both tables can be viewed on Yale image finder portal by editing the URL as http://sprout038.sprout.yale.edu/imagefinder/Figure.extern?sp=SPMC126015/1471-2164-3-21-4&state:Home=BrO0ABXcTAAAAAQAADHNIYXJjaFN0cmluZ3QABmNhbmNlcg%3D%3D.

### 3.2.3 Training of the OCR Algorithm

The first step to train the OCR algorithm requires seed images. TIFF image files with ‘Courier’ and ‘Times New Roman’ fonts were prepared as these are the mostly used fonts in sequence images [157]. Subsequently the JTessBoxEditor[^19] was invoked with the seed images to analyze the image components and to generate the coordinates of the characters. A training image should be at least $300 \times 300$ pixels and should at least be 1 bit per pixel encoded as UTF-8 format to yield the precise output. Figure 3.7 and Figure 3.8 depict the seed image and the image component analysis phases respectively.

[^19]: http://vietocr.sourceforge.net/training.html
Figure 3.7: The jTessBoxEditor view with Input File (Training Step -1)

jTessBoxEditor is a box editor and trainer for Tesseract OCR, providing editing of box data of both Tesseract 2.0x and 3.0x formats and full automation of Tesseract training. A seed image file with name sebi_training.tif was prepared and mounted on jTessBoxEditor as displays in Figure 3.7. In order to capture the special characters from images later in testing process, seed image incorporates maximum number special characters while training.

The jTessBoxeditor allows a user to split or merge the words/characters and to correct their coordinates (positions) with respect to the document. In some cases, it is necessary to manually correct the box file for images during the training process. A screenshot of box file editing process is displayed in the Figure 3.8.
Figure 3.8: A Component Analysis View of Training File in jTessBoxEditor

The component analysis view as displayed in Figure 3.8 provides the width, height and coordinate information about each character. One of the necessary steps before starting the training process is to correct a character’s information if it is defined imprecisely by the jTessBoxEditor.

In the final training stage, the jTessBoxTrainer generates a trained character dataset as a box file. The box file contains information about a character’s location with respect to the document or image, the character size, coordinates or dimensions and the scale information about all the characters in a TIFF seed file. The OCR algorithm further utilizes the box file at the production stage to detect and extract the optical characters from the images.
Figure 3.9: Box File Editing Process with jTessBoxEditor

[The advanced editing mode of the training file as shown in Figure 3.9 offers more options to edit the seed file such as merge, split, insert and delete. Moreover, a user can set margin and can scale a character with respect to other characters or document. Once the editing is finalized the jTessBoxEditor starts OCR algorithm training based on the seed image and generates a box file.]

A flowchart in Figure 3.10 explains the OCR process. We have set a minimum threshold value of fifteen characters as the target output. If the output of the OCR algorithm generates fifteen characters or less, it calls the image-acquisition algorithm (IAA) which connects with PubMed API to look for high resolution images. In case the IAA fails to find any image with high resolution than the previous one, it invokes the crowd annotator function where a user can annotate an image manually.
3.2.4 OCR Post-processing

The OCR process extracts all characters (dictionary words and clinical terminologies), digits, special characters from a sequence image. However, to generate the annotations using the sequence analysis resources the input should be free from dictionary words, clinical terminologies, digits
and special characters except the dash (-) that represents a gap in the sequence [158]. Therefore, JAWS, a Java API for WordNet [159], and the SNOMED-CT API [160] was used to detect and eliminate the clinical terminologies from the output. A Java class takes the OCR output as input and scans for digits, special characters, dictionary words and clinical terminologies. An algorithm has been designed that removes the digits and unwanted special characters at the first stage and forwards the outcomes to the second stage where two separate functions transform the sequence characters (output) into separate strings by delimiting the sequence block based on spaces. Essentially, two web services that hold the functionality of WordNet and SNOMED-CT have been invoked to identify and to eliminate all the dictionary and clinical terminologies from the input. The result of this process yields a clean input ready for submission to a sequence analysis web service that later produces the annotations to enrich the sequence image. The sections below explain the functionalities of WordNet, JAWS and SNOMED-CT.

3.2.5 WordNet and SNOMED CT

WordNet is a large collation of lexical English language data that has been stored electronically. WordNet covers English nouns, verbs, adjectives and adverbs. Unlike a classic dictionary, where a page holds a number of words with different meanings, WordNet arranges the words that share some lexical similarity (word to word) or semantic similarity (concept to concept). Essentially, the WordNet arranges the words into groups called as synsets where each synset describes a distinct concept. Since the WordNet is freely available (under a BSD style license) and can be accessed programmatically, it has been utilized widely in text analysis, document sorting and retrieval [158]. A Java based dictionary is available to manipulate the text data; the API is called JAWS (Java API for WordNet Searching). JAWS is compatible with Java 5 and later versions and come with several valuable classes and methods that speed up the text analysis process. In order to detect, extract and
manage the dictionary words from the sequence string, the WordNet API was wrapped as a web service that can automatically connect to the database to acquire the input, perform dictionary checking and save the output back into the database [158]. SNOMED CT is recognized as the most comprehensive list of clinical terminologies, and is available to date in multiple languages. SNOMED CT plays a vital role in the development process of meaningful patient health records that further streamline the overall treatment process.

3.2.6 Global and Local Level Classification of Sequence Images

The post processing process on the sequence OCR removes all the medical terms and dictionary words from OCR strings. This refined OCR is passed to the DIA and SISA. DIA connects with a biomedical image repository and processes the images one by one in order to detect images with non-dictionary words. Technically, the algorithms take the sequence strings and matches them with a dictionary and medical term list. This process selects all the sequence images as the sequence images hold non-dictionary characters. The algorithm annotates the retrieved sequence images as ‘sequence’ and the rest of the images as ‘non-sequence’. Similarly, to classify sequence images between protein and DNA sequence images, the algorithm traverses through the sequence string and examines each character against the DNA ASCII string. Once the algorithm finds an image that holds only the characters ATCG, it annotates that sequence as a DNA image and annotates the rest of the sequence images as protein sequences. The algorithm in Figure 3.11 describes the sequence segregation algorithm graphically.
**Algorithm 1** Sequence Images Segregation Algorithm

1: **procedure** RECOGNITION AND EXTRACTION OF SEQUENCE IMAGES
2:   Let $X_{img}$ is an array of images’ optical characters
3:   $X_{img}[] = \{x_1, x_2, x_3, \ldots, x_n\}$
4:   Let $Y$ is an array of of English alphabets from A to Z
5:   $Y[] = \{A - Z\}$
6:   Let $Z$ is an array with DNA base pairs
7:   $Z[] = \{A, T, C, G\}$
8:   **for** $X_{img}$ each node $i \in N$ **do**
9:       if ($X_{img}[i].eachCharacter() == Y$
10:         and ($X_{img}[i] != dictionary word$)) **then**
11:          add property with $X_{img}[i]$ as ”Sequence Image”
12:          store.initDB(”SequenceImage”)
13:       **else**
14:          store.initDB(”Non - SequenceImage”)
15:   **Endfor**

16: Extraction of Protein and DNA Sequence Images
17:   Let $Sequence_{ia}$ is an array of sequence image ASCII
18:   $Sequence_{ia}[] = \{s_1, s_2, s_3, \ldots, s_n\}$
19:   **for** $Sequence_{ia}$ each node $i \in N$ **do**
20:       if ($Sequence_{ia}[i].eachCharacter() == z[i]$) **then**
21:          add property with $X_{img}[i]$ as ”DNA Sequence Image”
22:          store.initDB(”DNASequenceImage”)
23:       **else**
24:          store.initDB(”ProteinSequenceImage”)
25:   **Endfor**

26: **Endprocedure**

Figure 3.11: The DNA and Protein Sequence Segregation Algorithm (SISA)
3.3 Results and Evaluation
This section provides details of the results and evaluations of SEBI sequence image training and testing with different image resolutions. Overall the performance evaluations have produced favorable results that demonstrate the legitimacy of the proposed approach.

3.3.1 Performance Evaluation of OCR Algorithm

This section has been designed to illustrate the testing and evaluation of the SEBI OCR algorithm. To test the accuracy of the algorithm, two hundred images were selected randomly from the sequence image repository. It is important to note that training and testing repositories hold two different images datasets. We conducted the experiments and recorded the results with four different scenarios. At first, the OCR algorithm was initialized with images of mixed resolution and an untrained algorithm for which the character level accuracy and job completion time are recorded in Table 3.1. Experiments were conducted with processed but low quality images (200*200 pixels) using the trained the OCR engine as well as the high quality unprocessed images using the trained OCR engine. The recorded the character level accuracy and job completion time are shown in Table 3.2 and Table 3.3 respectively.

Input images were processed with multiple image filters (See section 3.2.4) to increase the resolution and contrast. Prior to applying the OCR algorithm on high quality processed images, the OCR algorithm was also trained with the font-specific (Courier and Times New Roman) datasets. Since a biological sequence string does not hold any dictionary words, the default dictionary check feature of Tesseract engine [152] was set to off to make the extraction process fast. The results of processed images with a fully trained OCR engine (with font-specific training) can be viewed in Table 3.4. Details of the application of image filters on image processing (pre and post) can be studied in section 7.2. To analyze the change in the performance level of the OCR engine, images in the batches were swapped out and experiments were repeated. At the beginning,
the twenty test images in a batch were formulated with five low, five medium and five high-level resolution images, along with five multi-page images where the sequence strings reside in some section of the image.

3.3.2 Measurement of Character Level Accuracy (CLA)

To judge the performance of the OCR detection and extraction algorithm, a character-level accuracy (CLA) measure was applied to the output generated by the OCR algorithm. The CLA divides the character errors per batch by the total number of characters per batch and subtracts the results from 1 prior to multiplying it by 100 to get the percentage score of OCR algorithm accuracy. Further details of character-level accuracy can be studied in [28] where it was introduced to compare the performance of Tesseract and ABBYY FineReader\textsuperscript{20} OCR engines.

\[
CLA = \left(1 - \frac{\text{Character Errors Per Batch}}{\text{Total Characters Per Batch}}\right) \times 100
\]

The whole process of calculating CLA was carried out manually. While documenting the CLA results, multiple types of character error were observed; in particular, there were cases where the OCR algorithm represented a character twice, did not recognize a particular character, or left the character space blank in the output. Moreover, in certain cases the OCR engine falsely recognized a character. Therefore, we considered all the scenarios as OCR engine error. Tables in section 3.3.2 display the CLA score with different scenarios, as described in table legends. The signatures of training and testing datasets are listed in Table 3.6 and 3.7 where sequence images can be viewed and downloaded from Yale image finder portal by editing the URL as http://sprout038.sprout.yale.edu/imagefinder/Figure.external?sp=SPMC126015/1471-2164-3-21-4&state:Home=Br00ABXcTAAAAAQADHNIYXJaFN0cmuZ3QABmNhbnNlec%3D%3D.

\textsuperscript{20} http://www.abbYY.com/finereader/
Table 3.1: Character Level Accuracy with unprocessed images and untrained algorithm

[In Table 3.1 displays the CLA score with unprocessed/raw sequence images and with untrained OCR algorithm. 69.5% is the average accuracy score that was calculated in this case. The datasets names column represents the name and number of datasets that have been used in the OCR test process. Column 2 shows the number of images per batch; we formulated each batch with 20 images. Characters per batch and character error per batch were calculated semi-automatically. JCT is the job completion time in seconds per batch, which we gathered from query log file. The last column is the character level accuracy that we calculated with the formula mentioned in section 3.3.2]

<table>
<thead>
<tr>
<th># Test Images</th>
<th>Characters Per Batch</th>
<th>Character Error Per Batch</th>
<th>JCT*</th>
<th>Character Level Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch - 1</td>
<td>20</td>
<td>1749</td>
<td>500</td>
<td>840</td>
</tr>
<tr>
<td>Batch - 2</td>
<td>20</td>
<td>2080</td>
<td>652</td>
<td>360</td>
</tr>
<tr>
<td>Batch - 3</td>
<td>20</td>
<td>952</td>
<td>152</td>
<td>540</td>
</tr>
<tr>
<td>Batch - 4</td>
<td>20</td>
<td>4026</td>
<td>1874</td>
<td>480</td>
</tr>
<tr>
<td>Batch - 5</td>
<td>20</td>
<td>1371</td>
<td>329</td>
<td>540</td>
</tr>
<tr>
<td>Batch - 6</td>
<td>20</td>
<td>6527</td>
<td>4524</td>
<td>900</td>
</tr>
<tr>
<td>Batch - 7</td>
<td>20</td>
<td>1209</td>
<td>225</td>
<td>780</td>
</tr>
<tr>
<td>Batch - 8</td>
<td>20</td>
<td>5563</td>
<td>764</td>
<td>1380</td>
</tr>
<tr>
<td>Batch - 9</td>
<td>20</td>
<td>693</td>
<td>253</td>
<td>900</td>
</tr>
<tr>
<td>Batch - 10</td>
<td>20</td>
<td>2316</td>
<td>492</td>
<td>1140</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>200</strong></td>
<td><strong>26486</strong></td>
<td><strong>9765</strong></td>
<td><strong>7860</strong></td>
</tr>
</tbody>
</table>

*JCT= Job completion time in seconds*
Table 3.2: Character Level Accuracy with pre-processed but low quality images

<table>
<thead>
<tr>
<th>Batch</th>
<th># Test Images</th>
<th>Characters Per Batch</th>
<th>Character Error Per Batch</th>
<th>JCT*</th>
<th>Character Level Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>1749</td>
<td>395</td>
<td>779</td>
<td>77.5</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>2080</td>
<td>531</td>
<td>302</td>
<td>74.5</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>952</td>
<td>129</td>
<td>511</td>
<td>86.5</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>4026</td>
<td>1498</td>
<td>465</td>
<td>62.8</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>1371</td>
<td>300</td>
<td>439</td>
<td>78.2</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>6527</td>
<td>3598</td>
<td>892</td>
<td>44.9</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>1209</td>
<td>169</td>
<td>670</td>
<td>86.1</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>5563</td>
<td>744</td>
<td>1090</td>
<td>86.7</td>
</tr>
<tr>
<td>9</td>
<td>20</td>
<td>693</td>
<td>265</td>
<td>884</td>
<td>61.8</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>2316</td>
<td>492</td>
<td>892</td>
<td>78.8</td>
</tr>
<tr>
<td>Total</td>
<td>200</td>
<td>26486</td>
<td>9765</td>
<td>6924</td>
<td>73.8</td>
</tr>
</tbody>
</table>

*JCT= Job completion time in seconds
Table 3.3: Character Level Accuracy with high quality unprocessed images

[In Table 3.3, provides detail about the character level accuracy of OCR algorithm with high quality unprocessed images along with trained OCR algorithm. The dataset names represent the name and number of datasets that have been used in the OCR test process. Column 2 shows the number of images per batch, we formulated each batch with 20 images. Characters per batch and character error per were calculated semi-automatically. JCT is the job completion time in seconds per batch, which we have gathered from query log file. The last column is the character level accuracy that we have calculated with the formula mentioned in section 3.3.2]

<table>
<thead>
<tr>
<th>Batch</th>
<th># Test Images</th>
<th>Characters Per Batch</th>
<th>Character Error Per Batch</th>
<th>JCT*</th>
<th>Character Level Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch - 1</td>
<td>20</td>
<td>1749</td>
<td>219</td>
<td>668</td>
<td>87.5</td>
</tr>
<tr>
<td>Batch - 2</td>
<td>20</td>
<td>2080</td>
<td>439</td>
<td>265</td>
<td>78.9</td>
</tr>
<tr>
<td>Batch - 3</td>
<td>20</td>
<td>952</td>
<td>89</td>
<td>502</td>
<td>90.7</td>
</tr>
<tr>
<td>Batch - 4</td>
<td>20</td>
<td>4026</td>
<td>1299</td>
<td>451</td>
<td>67.8</td>
</tr>
<tr>
<td>Batch - 5</td>
<td>20</td>
<td>1371</td>
<td>228</td>
<td>403</td>
<td>83.4</td>
</tr>
<tr>
<td>Batch - 6</td>
<td>20</td>
<td>6527</td>
<td>955</td>
<td>663</td>
<td>85.4</td>
</tr>
<tr>
<td>Batch - 7</td>
<td>20</td>
<td>1209</td>
<td>110</td>
<td>579</td>
<td>90.9</td>
</tr>
<tr>
<td>Batch - 8</td>
<td>20</td>
<td>5563</td>
<td>402</td>
<td>1002</td>
<td>92.8</td>
</tr>
<tr>
<td>Batch - 9</td>
<td>20</td>
<td>693</td>
<td>163</td>
<td>701</td>
<td>76.5</td>
</tr>
<tr>
<td>Batch - 10</td>
<td>20</td>
<td>2316</td>
<td>176</td>
<td>670</td>
<td>92.5</td>
</tr>
<tr>
<td>Total</td>
<td>200</td>
<td>26486</td>
<td>4952</td>
<td>5904</td>
<td>84.7</td>
</tr>
</tbody>
</table>
Table 3.4: Character Level Accuracy with high quality processed images

In Table 3.4 provides statistics about CLA with high quality processed images along with font-specific trained OCR engine. The dataset names represent the name and number of datasets that were used in OCR test process. Column 2 shows the number of images per batch, we formulated each batch with 20 images. Characters per batch and character error per batch were calculated semi-automatically. JCT is the job completion time in seconds per batch, which we have gathered from query log file. The last column is the character level accuracy that we have calculated with the formula mentioned in section 3.3.2.

<table>
<thead>
<tr>
<th>Batch</th>
<th># Test Images</th>
<th>Characters Per Batch</th>
<th>Character Error Per Batch</th>
<th>JCT*</th>
<th>Character Level Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch - 1</td>
<td>20</td>
<td>1749</td>
<td>30</td>
<td>329</td>
<td>98.3</td>
</tr>
<tr>
<td>Batch - 2</td>
<td>20</td>
<td>2080</td>
<td>20</td>
<td>184</td>
<td>99.1</td>
</tr>
<tr>
<td>Batch - 3</td>
<td>20</td>
<td>952</td>
<td>22</td>
<td>492</td>
<td>97.7</td>
</tr>
<tr>
<td>Batch - 4</td>
<td>20</td>
<td>4026</td>
<td>19</td>
<td>436</td>
<td>99.6</td>
</tr>
<tr>
<td>Batch - 5</td>
<td>20</td>
<td>1371</td>
<td>30</td>
<td>368</td>
<td>97.9</td>
</tr>
<tr>
<td>Batch - 6</td>
<td>20</td>
<td>6527</td>
<td>52</td>
<td>324</td>
<td>99.3</td>
</tr>
<tr>
<td>Batch - 7</td>
<td>20</td>
<td>1209</td>
<td>32</td>
<td>485</td>
<td>97.4</td>
</tr>
<tr>
<td>Batch - 8</td>
<td>20</td>
<td>5563</td>
<td>13</td>
<td>965</td>
<td>99.8</td>
</tr>
<tr>
<td>Batch - 9</td>
<td>20</td>
<td>693</td>
<td>9</td>
<td>785</td>
<td>98.8</td>
</tr>
<tr>
<td>Batch - 10</td>
<td>20</td>
<td>2316</td>
<td>28</td>
<td>395</td>
<td>98.8</td>
</tr>
<tr>
<td>Total</td>
<td>200</td>
<td>26486</td>
<td>255</td>
<td>4763</td>
<td>98.6</td>
</tr>
</tbody>
</table>

*JCT= Job completion time in seconds

3.3.3 Character Level Accuracy Comparison

With the default optimization level, the OCR algorithm yielded 9725-character level errors out of 26486 characters. Figure 3.12 displays a character level accuracy comparison with both processed and unprocessed images, with a trained OCR engine with a default setting, and with a font-specific
trained OCR algorithm. Here the comparison is based on both single and multiple sequence alignment images. It can be observed that the accuracy of the OCR engine decreases with an increase of low resolution images. The lowest accuracy score recorded with an image set of resolution < 200*200 was 69.5%. Experiments also show that the popular image formats commonly used on the web such as gif and jpeg do not yield high quality OCR output.

![Character Level Accuracy Comparison](image)

**Figure 3.12: Character Level Accuracy Comparison**
With pre-processed images the OCR algorithm yielded an overall accuracy of 98.6%. A font-specific training and image PRE-processing is required to achieve the maximum character level accuracy score. It is important to note that 98.6% accuracy was only possible when the data was of high quality and the OCR algorithm was fully trained with courier fonts. Overall, we achieved accuracy (CLA) within the range of 69.5% ~ 98.6%. While performing the experiments, we observed that the overall job processing time was reduced from 7860 to 4763 seconds with processed images and with a fully trained OCR engine. Figure 3.13 explains the job completion time per batch with all types of images.

Figure 3.13: A Graphical Comparison of Job Completion Time

[Figure 3.13 depicts the OCR extraction time in seconds. The graph clearly demonstrates the decline in time with the increase of preprocessing of images and with the training levels of OCR engine]
To summarize, the quality of the OCR is of utmost importance to the generation of accurate annotations using the sequence analysis web services. With a series of optimization experiments and by changing the multiple parameters, we were able to achieve a reasonable level of accuracy. Table 3.5 displays the training and testing statistics of the OCR algorithm.

**Table 3.5: Summary of OCR Algorithm Training and Testing**

*Table 3.5 displays both the experiments and evaluation statistics for optical character recognition algorithm. The average character level accuracy falls between 69.5 ~ 98.6% with different scenarios.*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Training Datasets</td>
<td>6</td>
</tr>
<tr>
<td>Number of Images used in Training</td>
<td>540</td>
</tr>
<tr>
<td>Type of Images In Training Datasets</td>
<td>Protein and DNA Sequence</td>
</tr>
<tr>
<td>Low Resolution Images In Training Dataset</td>
<td>250</td>
</tr>
<tr>
<td>Medium to High Resolution Images In Training Dataset</td>
<td>150</td>
</tr>
<tr>
<td>Sequence Images with in other image types</td>
<td>140</td>
</tr>
<tr>
<td>Number of Images used in Testing</td>
<td>200</td>
</tr>
<tr>
<td>Average Character Level Accuracy with normal images and without training the algorithm</td>
<td>69.5%</td>
</tr>
<tr>
<td>Average Character Level Accuracy with processed but low quality images (200* 200) and with trained OCR engine</td>
<td>73.8%</td>
</tr>
<tr>
<td>Average Character Level Accuracy with high quality unprocessed images and with fully trained OCR engine</td>
<td>84.6%</td>
</tr>
<tr>
<td>Average Character Level Accuracy with high quality processed images and with fully trained OCR engine</td>
<td>98.6%</td>
</tr>
<tr>
<td>Table 3.6: Image Signatures for the Datasets used in SEBI OCR Training</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>PMC1074309/gki337f3</td>
<td>PMC150437/gb-2001-2-1</td>
</tr>
<tr>
<td>PMC1074398/gki343f4</td>
<td>PMC1508164/1471-2148-6-42-3</td>
</tr>
<tr>
<td>PMC1079787/pbio.0030189.g001</td>
<td>PMC151178/gb-2002-3-12</td>
</tr>
<tr>
<td>PMC1079803/1471-2121-6-15-2</td>
<td>PMC151182/gb-2002-3-12</td>
</tr>
<tr>
<td>PMC1079851/1471-2180-5-13-1</td>
<td>PMC151187/gb-2002-3-12</td>
</tr>
<tr>
<td>PMC1079889/1472-6807-5-6-5</td>
<td>PMC151194/gb-2002-3-12</td>
</tr>
<tr>
<td>PMC1079965/gki349f2</td>
<td>PMC1513071/nrs04017.f1</td>
</tr>
<tr>
<td>PMC1079968/gki509f8</td>
<td>PMC1513258/1471-2105-7-213-3</td>
</tr>
<tr>
<td>PMC1079969/gki510f1</td>
<td>PMC1523194/1471-2156-7-34-1</td>
</tr>
<tr>
<td>PMC1084323/gki524f3</td>
<td>PMC1524780/1471-2091-7-19-2</td>
</tr>
<tr>
<td>PMC1084334/pbio.0030151.g002</td>
<td>PMC1525208/1471-2105-7-160-1</td>
</tr>
<tr>
<td>PMC1084347/1471-2180-5-9-2</td>
<td>PMC1526463/pcbi.0020075.g005</td>
</tr>
<tr>
<td>PMC1087835/1471-2105-6-93-4</td>
<td>PMC1534069/1471-2105-7-292-4</td>
</tr>
<tr>
<td>PMC1087892/1743-422X-2-33-3</td>
<td>PMC153517/1471-2148-3-4-2</td>
</tr>
<tr>
<td>PMC1087902/gki548f3</td>
<td>PMC1538770/gkl227f1</td>
</tr>
<tr>
<td>PMC1088945/gb-2005-6-3-r26-12</td>
<td>PMC1538770/gkl227f2</td>
</tr>
<tr>
<td>PMC1088965/gb-2005-6-4-r37-2</td>
<td>PMC1538791/gkl164f1</td>
</tr>
<tr>
<td>PMC1088969/gki537f1</td>
<td>PMC1538791/gkl164f2</td>
</tr>
<tr>
<td>PMC1097723/1471-2121-6-23-1</td>
<td>PMC1538800/gkl080f1</td>
</tr>
<tr>
<td>PMC1110740/gki564f2</td>
<td>PMC1538806/gkl171f3</td>
</tr>
<tr>
<td>PMC1110908/pbio.0030172.g002</td>
<td>PMC1538856/gkl159f2</td>
</tr>
<tr>
<td>PMC112587/1471-213X-5-8-2</td>
<td>PMC1538871/gkl330f4</td>
</tr>
<tr>
<td>PMC1126904/gki574f3</td>
<td>PMC1538996/1471-2121-7-24-4</td>
</tr>
<tr>
<td>PMC1126904/gki574f5</td>
<td>PMC1540707/pbio.0040269.g006</td>
</tr>
<tr>
<td>PMC113743/1471-2091-3-7-1</td>
<td>PMC1540718/gkl395f5</td>
</tr>
<tr>
<td>PMC1140083/gki613f2</td>
<td>PMC1540722/gkl459f8</td>
</tr>
<tr>
<td>PMC1142214/ijbsv01p0067g02</td>
<td>PMC1540725/gkl476f1</td>
</tr>
<tr>
<td>PMC1145181/1471-2148-5-31-1</td>
<td>PMC1543647/1471-2199-7-21-4</td>
</tr>
<tr>
<td>PMC1148173/gki651f2</td>
<td>PMC1550403/1471-2164-7-39-1</td>
</tr>
<tr>
<td>PMC1156898/1471-2199-6-12-2</td>
<td>PMC1552070/1471-2164-7-197-5</td>
</tr>
<tr>
<td>PMC1157100/gki676f2</td>
<td>PMC1552076/1746-1448-2-10-1</td>
</tr>
<tr>
<td>PMC1159119/gki672f2</td>
<td>PMC1555579/1471-2121-7-31-1</td>
</tr>
<tr>
<td>PMC101403/1471-2091-3-5-5</td>
<td>PMC1489923/1745-6150-1-12-2</td>
</tr>
<tr>
<td>PMC102321/1471-2148-2-3-3</td>
<td>PMC1489923/1745-6150-1-12-5</td>
</tr>
<tr>
<td>PMC102765/1471-2164-3-9-1</td>
<td>PMC149452/1471-2180-3-1-2</td>
</tr>
</tbody>
</table>

90
<table>
<thead>
<tr>
<th>PMID</th>
<th>Reference</th>
<th>PMID</th>
<th>Reference</th>
<th>PMID</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMC1487177/pgen.0020099.g002</td>
<td>PMC1853101/1471-2180-7-25-4</td>
<td>PMC2132714/JCB29394.f5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMC1462995/1471-2180-6-36-1</td>
<td>PMC1838911/1471-2229-7-13-3</td>
<td>PMC2118257/jem2030699f06</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMC1464127/1471-2164-7-83-1</td>
<td>PMC184357/1471-2148-3-17-1</td>
<td>PMC2118641/jem2041571f03</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMC1464399/ppat.0020050.g002</td>
<td>PMC1847441/1471-2148-7-43-1</td>
<td>PMC2121648/1471-2148-7-178-3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMC1464423/age_1409_f3</td>
<td>PMC1847502/imb0015-0687-f4</td>
<td>PMC212514/1471-2164-4-34-5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMC1484252/gkl415f8</td>
<td>PMC1852406/gb-2007-8-2-r29-1</td>
<td>PMC2132707/JCB33004.f8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.7: Image Signatures for the Dataset used in SEBI OCR Testing

<table>
<thead>
<tr>
<th>Image Signatures</th>
<th>PMCID</th>
<th>DOAJ Digital Object Identifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMC2132775/JCB9801086.f12</td>
<td>PMC2171356/200502113f1</td>
<td>PMC2194728/1471-2164-8-407-4</td>
</tr>
<tr>
<td>PMC2132778/JCB15193.f2</td>
<td>PMC2171420/200503106f6</td>
<td>PMC2194796/1471-2105-8-307-6</td>
</tr>
<tr>
<td>PMC2132829/JCB9807002.f5a</td>
<td>PMC2171497/200411047f7</td>
<td>PMC2195581/JEM990333.f1a</td>
</tr>
<tr>
<td>PMC2132843/JCB9805122.f3</td>
<td>PMC2171781/200411026f4</td>
<td>PMC2196299/JEM.galocha2</td>
</tr>
<tr>
<td>PMC2132904/JCB9809116.f1</td>
<td>PMC2172142/200310098f3a</td>
<td>PMC2196330/JEM.5545f6e</td>
</tr>
<tr>
<td>PMC2132977/JCB9807048.f1a</td>
<td>PMC2172378/200403011f6</td>
<td>PMC2196334/JEM.962176f1</td>
</tr>
<tr>
<td>PMC2133011/JCB32948.f7a</td>
<td>PMC2172406/200403174f2</td>
<td>PMC2196340/JEM.5550f1</td>
</tr>
<tr>
<td>PMC2133028/JCB29472.f5a</td>
<td>PMC2172454/200406048f1</td>
<td>PMC2196448/0105089f2</td>
</tr>
<tr>
<td>PMC2133037/JCB9803053.f4</td>
<td>PMC2172596/200408109f3</td>
<td>PMC2196875/0102110f2</td>
</tr>
<tr>
<td>PMC2133037/JCB9803053.f5</td>
<td>PMC2172649/200208109f2</td>
<td>PMC2196876/0102142f1a</td>
</tr>
<tr>
<td>PMC2133068/JCB9808025.f2</td>
<td>PMC2172718/200212136f1</td>
<td>PMC2198805/JCB0008148.f1</td>
</tr>
<tr>
<td>PMC2133194/JCB9812064.f3a</td>
<td>PMC2172798/200301046f1</td>
<td>PMC2198836/0107126f9</td>
</tr>
<tr>
<td>PMC2134285/JCB.halverson6</td>
<td>PMC2172854/200302129f1</td>
<td>PMC2199003/JEM.970608f1</td>
</tr>
<tr>
<td>PMC2134290/JCB.kim1</td>
<td>PMC2172915/200210130f1</td>
<td>PMC2199019/JEM.970967f3a</td>
</tr>
<tr>
<td>PMC2136225/JCB.14475f3a</td>
<td>PMC2173024/200108145f1a</td>
<td>PMC2199049/JEM.970694f1</td>
</tr>
<tr>
<td>PMC2136226/JCB.patel4a</td>
<td>PMC2173100/200204102f2</td>
<td>PMC2199054/JEM.970796f5a</td>
</tr>
<tr>
<td>PMC2137815/JCB.14555f9</td>
<td>PMC2173206/200204063f8a</td>
<td>PMC2199077/JEM.971274f1a</td>
</tr>
<tr>
<td>PMC2138191/JCB.32751f2</td>
<td>PMC2173254/0109010f5</td>
<td>PMC2199107/JEM.971038f3</td>
</tr>
<tr>
<td>PMC2138193/JCB.32720f1</td>
<td>PMC2173464/0201081f1</td>
<td>PMC2199189/JEM.971948f1</td>
</tr>
<tr>
<td>PMC2139795/JCB.32844f2a</td>
<td>PMC2173547/0201048f1</td>
<td>PMC2199222/0108030f8</td>
</tr>
<tr>
<td>PMC2139813/JCB.32883f6</td>
<td>PMC2173682/200304112f7</td>
<td>PMC2199737/JCB9902050.f2</td>
</tr>
<tr>
<td>PMC2139890/JCB.mcdonald3</td>
<td>PMC2173741/200211113f1</td>
<td>PMC2199742/JCB9903019.f10</td>
</tr>
<tr>
<td>PMC2139943/JCB.32806f7</td>
<td>PMC2173857/0202047f4</td>
<td>PMC2199744/JCB9904126.f1</td>
</tr>
<tr>
<td>PMC2140092/11103_2007_9257_Fig1_HTML</td>
<td>PMC2173965/200307089f2</td>
<td>PMC2204015/1471-2164-8-391-5</td>
</tr>
<tr>
<td>PMC2140169/JCB32972.f7</td>
<td>PMC2174006/200204081f4</td>
<td>PMC2206062/1471-2105-8-417-5</td>
</tr>
<tr>
<td>PMC2140216/JCB.10996f1c</td>
<td>PMC2174032/0204039f2</td>
<td>PMC2206118/7601936f4</td>
</tr>
<tr>
<td>PMC2141637/JCB.32774f2</td>
<td>PMC2174084/0110063f1</td>
<td>PMC2211285/1742-6405-4-24-4</td>
</tr>
<tr>
<td>PMC2141742/JCB15030.f2a</td>
<td>PMC2174219/JCB9904131.f6</td>
<td>PMC2211313/1471-2148-7-199-2</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>PMC2148117/JCB9805123.f1</td>
<td>PMC2174333/JCB0102016.f1</td>
<td>PMC2211546/pbio.0060007.g007</td>
</tr>
<tr>
<td>PMC2148128/JCB9808030.f1</td>
<td>PMC2174416/97320630002043F3</td>
<td>PMC2211552/ppat.0040008.g001</td>
</tr>
<tr>
<td>PMC2148134/JCB9808004.f2a</td>
<td>PMC2174749/1472-6807-7-62-1</td>
<td>PMC2212359/JEM980326.f2a</td>
</tr>
<tr>
<td>PMC2148174/JCB9804032.f3</td>
<td>PMC2174796/JCB9910111.f4</td>
<td>PMC2212410/JEM980831.f10</td>
</tr>
<tr>
<td>PMC2148182/JCB9803114.f1</td>
<td>PMC2174817/JCB0001127.f1</td>
<td>PMC2212531/JEM980617.f1a</td>
</tr>
<tr>
<td>PMC2148211/JCB9812072.f3</td>
<td>PMC2174859/JCB0002017.f2</td>
<td>PMC2212760/20041277f2</td>
</tr>
<tr>
<td>PMC2148225/JCB9812094.f1</td>
<td>PMC2174947/1471-2199-8-89-1</td>
<td>PMC2212834/20042031f3</td>
</tr>
<tr>
<td>PMC2148454/JCB33020.f2a</td>
<td>PMC2174952/1471-2148-7-166-4</td>
<td>PMC2212854/20050872f3</td>
</tr>
<tr>
<td>PMC2149341/JCB9804084.f5</td>
<td>PMC2175102/JCB9911080.f2</td>
<td>PMC2213183/20042510f4</td>
</tr>
<tr>
<td>PMC2150574/JCB9812024.f9b</td>
<td>PMC2175106/JCB9912077.f7</td>
<td>PMC2213434/JEM000304.f2</td>
</tr>
<tr>
<td>PMC2150581/JCB9812103.f1</td>
<td>PMC2175199/JCB0004117.f1</td>
<td>PMC2213584/jcb1800403f07</td>
</tr>
<tr>
<td>PMC2150668/JCB0005069.f1</td>
<td>PMC2175214/JCB9808005.f4</td>
<td>PMC2213590/jcb1800325f02</td>
</tr>
<tr>
<td>PMC2150673/JCB0004140.f6</td>
<td>PMC2175238/JCB9807041.f1</td>
<td>PMC2215773/pone.0001566.g001</td>
</tr>
<tr>
<td>PMC2150840/0012039f1a</td>
<td>PMC2175238/JCB9807041.f1</td>
<td>PMC2215773/pone.0001566.g002</td>
</tr>
<tr>
<td>PMC2150966/jem2043157f01</td>
<td>PMC2175284/JCB0001080.f7</td>
<td>PMC2215773/pone.0001566.g003</td>
</tr>
<tr>
<td>PMC2151067/1471-2334-7-123-3</td>
<td>PMC2175300/gkm703f1</td>
<td>PMC2215773/pone.0001566.g004</td>
</tr>
<tr>
<td>PMC2151080/1471-2156-8-72-2</td>
<td>PMC2175309/gkm742f3</td>
<td>PMC2215773/pone.0001566.g005</td>
</tr>
<tr>
<td>PMC2151134/pone.0001400.g001</td>
<td>PMC2180229/JCB0006076.f2</td>
<td>PMC2215773/pone.0001566.g006</td>
</tr>
<tr>
<td>PMC2151181/JCB9908001.f2</td>
<td>PMC2185566/JCB0002015.f4</td>
<td>PMC2216038/1471-2164-8-412-5</td>
</tr>
<tr>
<td>PMC2151507/jgp1270309f01</td>
<td>PMC2185576/JCB0005012.f5</td>
<td>PMC2216045/1471-2105-8-465-4</td>
</tr>
<tr>
<td>PMC2151562/jgp1280283f01</td>
<td>PMC2192391/JCB0012002.f6</td>
<td>PMC2216057/CFG2007-35604.002</td>
</tr>
<tr>
<td>PMC2151563/jgp1280293f01</td>
<td>PMC2192928/JEM981925.f3</td>
<td>PMC2217053/JGP.schild1</td>
</tr>
<tr>
<td>PMC2151571/jgp1280443f01</td>
<td>PMC2193037/JEM981895.f2</td>
<td>PMC2217218/JGP8139.f4</td>
</tr>
<tr>
<td>PMC2151584/jgp1280605f05</td>
<td>PMC2193370/JEM010159.f1a</td>
<td>PMC2217248/JGP8336.f7</td>
</tr>
<tr>
<td>PMC2151611/jgp1290245f01</td>
<td>PMC2193489/JEM010271.f4</td>
<td>PMC2217331/20028768f1</td>
</tr>
<tr>
<td>PMC2151615/jgp1290209f01</td>
<td>PMC2193584/011487f1</td>
<td>PMC2217355/200308788f1</td>
</tr>
<tr>
<td>PMC2151762/pone.0001402.g005</td>
<td>PMC2193610/011681f2a</td>
<td>PMC2217462/200309005f1</td>
</tr>
</tbody>
</table>
3.4 Discussion (of Sequence Image Processing Outcomes)

In SEBI, the semantic enrichment process begins by detecting and extracting the optical characters residing inside the sequence images. The sequence images are segregated from the raw image datasets. The raw image datasets contain various types of biomedical images. Details about biomedical images that are part of raw datasets can be studied in section 2.5. Moreover, sections 3.2.2 and 3.2.5 explain about the process of sequence images extraction and preprocessing. To optimize the resolution of the sequence image a number of image quality enhancer filters were employed (See section 3.2.1) and resolution was increased to at least 300 * 300 pixels. For the OCR detection and extraction process, we used 540 protein and DNA sequence images, dividing them into six training sets (See Table 3.6) based on their resolution. The OCR algorithm training is required to increase the algorithm’s efficiency and response time for the intended task. In 540 images, there were 250 images of with low resolution, and 150 images were selected manually with medium to high resolutions. Some images had fragment sequence strings displayed in different parts of the biomedical images. To train such images for OCR extraction, we included...
140 sequence images. *Tesseract 3.0, Leptonica* image and *ImageJ* libraries along with *jTessBoxEditor* GUI trainer were used to get the trained set of the sequence images.

We trained the OCR algorithm with courier and time new roman font sets and removed the other font families from the library. Moreover, we disabled the dictionary feature in Tesseract 3.0. Such actions significantly impacted the OCR algorithm’s overall efficiency. To observe the accuracy of OCR algorithms, we introduced character level accuracy scale, which provide a summarized overview of the accuracy. For the testing process, we used 200 randomly selected sequence images with different resolution and applied the OCR algorithms [28].

Our algorithm yielded 69.5% average accuracy with mixed images and an untrained algorithm and 73.8% average character level accuracy with processed but low quality images (200* 200) using a trained OCR engine. We achieved 84.6% and 98.6% average character level accuracy with high resolution unprocessed images along with a fully trained OCR engine and with high resolution processed images and a fully trained OCR engine respectively. Overall, the average character level OCR accuracy for our algorithm lay between 69.5 % ~ 98.6%. To yield the annotation from sequence ASCII from sequence analysis software, the input sequence string should be complete and correct. The purpose of this whole exercise of image preprocessing, training the OCR engine and doing the post processing was to achieve the maximum level of accuracy for the extraction of sequence string from the sequence image.
Chapter 4

Sequence Image Similarity and Classification

This chapter has been divided into introduction, methods, results and evaluation and discussion sections. The introduction section provides an overview of sequence image similarity and classification techniques in SEBI. The methods section highlights the semantic enrichment and similarity measurement procedures. The results and evaluation section provides experiment results and evaluation about SEBI overall performance. The last section summarizes the chapter and provides a commentary about SEBI results.

4.1 Introduction

The introduction section starts with the explanation of sequence image similarity measurement in the SEBI context. SEBI utilizes web services to generate annotations on sequence images that are further used to calculate the similarity among sequence images. Section 4.2.1 describes the available similarity measurement techniques and their limitations. Section 4.3 presents an overview of hard and soft image categorization techniques.
4.1.1 Sequence Image Similarity in SEBI Context

In SEBI, similarity between two sequence images is measured based on their derived annotations. SEBI utilizes the SADI web services [95] to generate annotations on sequence images. Details of SADI service modeling can be studied at sections 2.4.2, 4.2 and 4.3. Section 4.2.1 provides an overview of SADI services that have been developed to generate annotations on sequence images. Tier 2 of SEBI architecture (see Figures 1.1 and 1.2) explains the semantic enrichment process where optical characters are extracted from a sequence image and are passed to various sequence analysis services (see section 4.2.1) to generate new annotations. The image associated text such as caption and description are also mined to through BioNLP-SADI module [12] of SEBI to extract chemicals, drugs, Go Terms, species and proteins.

These generated annotations are further converted to RDF under the same semantic data model to achieve ad-hoc data integration among generated annotations. Section 4.3 explains the semantic modeling process of sequence image annotations by using the BIM ontology [23]. Previous approaches have used pattern matching techniques to find similar images [161]. However, sequence images are often identical in appearance, and by applying the pattern matching technique we cannot fetch the semantically similar images. Therefore, we have introduced sequence image similarity measurement criteria based on their derived semantic annotations. The two sequence images could be different visually; however, they could be similar based on their semantic annotations. Figure 4.1 illustrates sequence image similarity criteria graphically where two images (Sequence Image- 1 and Sequence Image -2) are different visually; however, they are similar to each other based on their semantic annotations.
4.1.2 Similarity Measurement Techniques

A number of techniques exist which measure the similarity between two or more given strings or patterns such as Euclidean distance [162], Dice coefficient [163], Jaccard similarity [164] and Overlap coefficient [165]. SEBI employs cosine similarity which is based on Vector Space Model (VSM) [21] to measure the similarity among sequence images’ annotations. Cosine similarity was selected for use in SEBI because of its documented accuracy relative to other available similarity measurement techniques [21]. Techniques such as Euclidean, Jaccard, Dice and overlap coefficient do not provide normalization criteria to handle term repetition, white spaces and special characters. The VSM was introduced in 1971 for the SMART information retrieval project [21]. During the development of the SMART information retrieval system, several new concepts and guidelines were established that are still being used in current search engines for information retrieval and knowledge representation [166]. VSM is essentially an algebraic model that represents textual information or documents in a corpus as a point(s) or vector(s) in a vector space. Points or vectors
that appear closer are considered related to each other and the points that appear far apart are regarded as distant [167]. An importance (a weight) is also assigned to the absence or presence of terms in a collection.

4.1.3 Sequence Image Classification

Clustering or categorization is a process to accumulate sets of similar objects into groups. Image clustering techniques are mainly divided into two types: hard clustering (i.e. K-Means) and soft clustering (i.e. Fuzzy C-Means). In hard clustering, each image belongs to exactly one cluster or group, whereas in soft clustering data elements can belong to multiple clusters [168]. To measure the similarity between sequence images we employed the Cosine similarity algorithm that computes the similarity score between zero and one. To classify the biomedical sequence images, cosine similarity has been paired with Fuzzy C-Means clustering in this thesis. Fuzzy C-Means is a powerful clustering technique that permits one piece of data to belong to multiple clusters or categorizes. Often the similarity algorithm in SEBI generates similarity score where images fall on boundaries of multiple categorizes and do not explicitly belong to a particular category or cluster. Therefore, the selection of Fuzzy C-Means clustering on K-Means [169] clustering was a reasonable choice [176]. Fuzzy C-Means clustering introduces the partial membership value (see section 2.7) which allows an image to associate itself with multiple categories [170].
4.2 Methods
The methods section has been divided into four sub-sections. Section 4.2.1 provides a list of web services, along with their description, that have been developed to generate annotations. SEBI generates annotations on sequence images by sending OCR outputs to sequence analysis and BioNLP as web services. The web services are semantically described by SADI, section 4.3 explains the semantic modeling process of sequence annotations that integrates the generated annotations semantically. Section 4.2.3 and 4.2.4 explain methodologies for similarity measurement and categorization in SEBI.

4.2.1 Annotation Generation by Using SADI Web Services

To measure the semantic similarity among sequence images, extracted annotations must be available in an integrated format. We have employed SADI web services to generate annotations on sequence images because they produce an integrated RDF annotation graph as output. A variety of SADI web services have been developed to generate annotations on sequence images. We categorize these SADI web services into three main categories based on the functions they perform. The operations performed by these web services includes BioNLP, sequence analysis, and image processing. The service name and description of generated SADI web services can be studied in section 2.4.2. Following are the examples of each web service category.

The BioNLP SADI web services take the image associated text, such as image captions and abstracts, and applies the BioNLP algorithms to find biological named entities such as drugs mentions. For instance, the first web service listed in table 4.1 is getDiseaseFromText is a BioNLP web service which scans the image associated text to look for the disease name.

The example of sequence analysis service is getPfamBySequenceString. This service takes the OCR extracted from the sequence images and finds the protein family to which a particular
sequence belongs. One of the examples of image processing service is `getImageResolutionByURL`. This provides a sequence image resolution in pixels by taking an image URL as input.

The outputs of these web services are consolidated automatically because all the services employ a common semantic model. Details of SADI web services and their input and output modeling can be studied at section 2.4.2. Moreover, the BioNLP annotation module normalizes or grounds the biological annotations with their structured resources available online such as UniProt [116] and DrugBank [7]. Table 4.1 lists the web services that have been designed to generate annotations on sequence images.

**Table 4.1: A SADI Registry of BioNLP and Sequence Analysis Web Services**

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Service Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getDiseaseFromText</code></td>
<td>This service extracts all possible diseases by receiving a text string as an input</td>
</tr>
<tr>
<td><code>getProteinFromText</code></td>
<td>With the text input, this service returns the protein mentioned in this text along with grounding resources</td>
</tr>
<tr>
<td><code>getChemicalFromText</code></td>
<td>This service takes the text input and mines it for chemical entities</td>
</tr>
<tr>
<td><code>getProteinInteractionFromText</code></td>
<td>This service finds the proteins, their interaction and growing information</td>
</tr>
<tr>
<td><code>getOrganismsFromText</code></td>
<td>This service provides the organism mentioned in the text by accepting the text input</td>
</tr>
<tr>
<td><code>getDrugsFromText</code></td>
<td>This service returns the drug names from text and grounds them with DrugBank</td>
</tr>
<tr>
<td><code>getGoTermsFromText</code></td>
<td>This service uses the GO API to find and extract the GO terms from text</td>
</tr>
<tr>
<td><code>getAbstractByDOI</code></td>
<td>This service takes document object identifiers as input and returns the corresponding abstract in RDF format.</td>
</tr>
<tr>
<td><code>getFulltextByPMCID</code></td>
<td>This service uses the PMCID of an article to find and return the full text of that article.</td>
</tr>
<tr>
<td>Service Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>getImagesByPMID</td>
<td>This service fetches biomedical images and their descriptions by taking PubMed IDs as input.</td>
</tr>
<tr>
<td>getImagesFromText</td>
<td>This service returns a biomedical image based on any keyword input.</td>
</tr>
<tr>
<td>getGeneAlterNames</td>
<td>This service returns alternate names for a gene name input; the input can be an abbreviated version of the gene name.</td>
</tr>
<tr>
<td>getMutationImagesByGeneID</td>
<td>This service extracts images of mutated genes by taking a Gene/Protein ID as input from iCyrus.</td>
</tr>
<tr>
<td>getOCRByImageURL</td>
<td>This service yields the image, OCR by getting the URL of an image as input.</td>
</tr>
<tr>
<td>getSequenceTypeByASCII</td>
<td>This service takes the ASCII string of a sequence and returns the sequence type such as: protein or DNA</td>
</tr>
<tr>
<td>getImageResolutionByURL</td>
<td>This service takes the image URL as an input and provides the image resolution information as an output.</td>
</tr>
<tr>
<td>getImagesLayoutByURL</td>
<td>This service provides the information about the present layout of an image</td>
</tr>
<tr>
<td>getDictionaryWordFromText</td>
<td>All the dictionary words can be retrieved by using the getDictionaryWordsFromText web service</td>
</tr>
<tr>
<td>getSnomedctFomText</td>
<td>This service detects and extracts the snomed-CD terms from the text input.</td>
</tr>
<tr>
<td>getBLASTpBySequenceString</td>
<td>This service generates the BLASTp hits with the input of a sequence string.</td>
</tr>
<tr>
<td>getBLASTnBySequenceString</td>
<td>This service generates the BLASTn hits with the input of a sequence string.</td>
</tr>
<tr>
<td>getpHMMERBySequenceString</td>
<td>HMMER protein sequence analysis can be perceived with the input of a sequence string.</td>
</tr>
<tr>
<td>getPfamBySequenceString</td>
<td>By taking a FASTA format sequence this service returns the protein family information.</td>
</tr>
<tr>
<td>getProteinClanByPfam</td>
<td>This service takes the protein family information as input and returns the protein clan information.</td>
</tr>
</tbody>
</table>
getPrositeScanBySequenceString | This service takes the FASTA format sequence string and generates the Prosite Scan output.

### 4.2.2 Semantic Modeling for the Sequence Image Annotations

BIM ontology is a domain-specific ontology that has been developed to provide the necessary semantic vocabularies for iCyrus, which is a SEBI module (see Figure 1.2). The rich hierarchy of semantic terms and relationships provided by the BIM ontology is further used to convert the syntactic annotations into RDF. Since the purpose of BIM ontology is to deliver the semantic terminologies for the RDF conversion, it was not required to add the logical/inference rules or complex axioms within the current version of BIM ontology. BIM ontology can be downloaded at [171] and details of BIM ontology can be studied at [23]. BIM provides four types of classes and properties to SEBI: 1) Annotation vocabularies i.e. `BIM:hasImageAnnotationSet`, `BIM:hasSequenceType`, `BIM:hasSequenceImage`, `BIM:hasNonSequenceImage`, `BIM:ProteinSequence`, `BIM:DNASequence` 2) Provenance Vocabularies i.e. `BIM:createdBy`, `BIM:createdOn` and `BIM:AnnVerifyBy` 3) Features and function vocabularies i.e. `hasSequenceMotif`, `hasConservedResidue` and 4) Semantic service vocabularies i.e. SADI services input and output classes. Figure 4.2 shows a BIM modeling to RDF conversion the sequence analysis annotations.

Figure 4.2 displays a sequence image annotation semantic modeling with BIM ontology. In Figure 4.2, there is an anonymous resource (dotted circle entity) that could be a scientific paper or book. This resource has an image which is of type sequence image. The sequence image here is semantically represented as `BIM:SequenceImage`, whereas the provenance of image has been defined as `BIM:ImageURL`. On the bottom of the sequence image, there is an annotation set which
is generated by a sequence analysis web service represented as \textit{BIM:WebService}. BIM also provides a provenance modeling for the sequence images and their related annotations. In Figure 4.2, the provenance vocabularies document their details about annotation creation date and curation resource as \textit{BIM: CreatedOn} and \textit{BIM:CreatedBy} and by annotation verification \textit{BIM:AnnVerifyBy}.

Figure 4.2: BIM Modeling to convert the Sequence Analysis Annotations into RDF
Figure 4.2 represents the semantic vocabularies for the SEBI sequence images enrichment process. The upper-boxed portion of the figure provides the semantic representations for an image provenance such as who created and verified the annotations and when the annotations were created. As the annotations can be created by a machine through web services, provenance part maintains the information about machine IP address and web service URL. The lower part of the figure highlights the semantic vocabularies used to convert the sequence analysis software outputs as an RDF such as BIM:hasMotif and BIM:hasMutationResidue.

4.2.3 Vector Space Model based Sequence Image Similarity Measurement

As stated in section 4.1.2, we employed the cosine similarity to calculate the similarity score among sequence images based on their derived annotations. The derived annotations after the enrichment process were stored as an RDF graph in a triple store. This section defines our methodology to apply the cosine similarity measure on the graph data. To demonstrate our approach, here we take a use case where a triplestore has three sequence images with their annotations presented as RDF graphs. Figure 4.3 presents a vector space diagram in which vectors of three images are presented as I-ag1, I-ag2 and I-ag3 respectively, I-ag stands for image annotation graph. A query graph has also been drawn as Q-ag. Q-ag has an annotation set such as hasDomain as "SH-2 Domain", hasUniProtID as "P98073", hasPfamily as "PDT" and hasDiseaseName as "Malaria". The I-ag1 and I-ag2 have three annotations the same as the Q-ag. However, both are lacking “hasDiseasesName”, the fourth annotation. I-ag3 has all four annotations similar to the Q-ag. To determine the most relevant graph, the cosine similarity technique calculates the angles between the query graph and rest of the annotations graph vectors. Therefore, the cosine similarity between I-ag3 and Q-ag or any two vectors is the measure of COS (Θ) between two vectors [172, 173] as displayed in Figure 4.3. The VSM is essentially an algebraic model that represents textual information or documents in a corpus as a point(s) or vector(s) in a vector space. Points or vectors that appear closer are considered related to each other, and the points that appear far apart are regarded as distant. VSM calculates the cosine of angle between query and other all annotations.
graphs and ranks the score from higher to low. The cosine of any two vectors can be calculated by applying the Euclidean dot product as:

\[
\cos(\theta) = \frac{\langle \hat{Q}-ag, \hat{I}-ag3 \rangle}{|\hat{Q}-ag||\hat{I}-ag3|}
\]

(1)

The unit vector can be derived by dividing a vector by its magnitude, as in case of Q-ag and I-ag3:

\[
\frac{Q-ag}{|Q-ag|} \cdot \frac{I-ag3}{|I-ag3|}
\]

(2)

Equation 3 represents the dot product of two unit vectors. For any two normalized vectors, cosine similarity is basically the dot or scalar product between two vectors.

\[
= \hat{Q}-ag \cdot \hat{I}-ag
\]

(3)

The distribution of concepts in a query vector and distribution of concepts in an I-ag3 vector are very similar according to the diagram. However, the Euclidean distance between query vector and
I-ag3 vector is relatively large. It shows that the vector I-ag3 is not in a normalized form. Equation 4 is used to normalize the length of I-ag3 [174].

\[
\frac{\sum_{i=1}^{n}(Q - a_{g_i}) (I - a_{g_i})}{\sqrt{\sum_{i=1}^{n}(Q - a_{g_i})^2} \sqrt{\sum_{i=1}^{n}(I - a_{g_i})^2}}
\]

Here Q-agi is the *nf-igf (Node frequency–inverse graph frequency)* [174] weight of *ith term* in the query and I-agi is the *nf-igf* weight of *ith term* in the I-ag3. *nf–igf* is a numerical statistic that is intended to reflect how important a node is to a graph in a collection or triplestore. The cosine similarity records the similarity score between 0 and 1. Section 4.2.4 utilizes these readings and applies the fuzzy based rules for the sequence image categorization.

### 4.2.4 Sequence Image Classification Rules

This section lists the inference rules that the SEBI fuzzy C-Means algorithm applies to categorize the sequence images into identical, similar, partial similar, partial distinct and distinct categories. Fuzzy rules are based on a series of IF and THEN statements with no ELSE in the rules set. Table 4.2 presents the fuzzy rule matrix where the CMF stands for *Concept Matching Frequency*, whereas CSM is the *Cosine Similarity Measure*. Notations VH, H, M, L, VL represent the relatedness intensity as Very High, High, Medium, Low and Very Low [150].
The inference engine in fuzzy C-Means [168] algorithm uses the concept matching frequency (CMF) and cosine similarity values to categorize the sequence images into different categories. Following are the categorization rules that SEBI framework utilizes to categorize the sequence images based on their semantic similarity score. M, H, VH, L, VL are the linguistics variables, and their value have been assigned through fuzzy membership function.

Rule 1: IF (CMF is VH and CSM is VH) then Images are Identical
Rule 2: IF (CMF is VH and CSM is H) then Images are Similar
Rule 3: IF (CMF is VH and CSM is M) then Images are Partially Similar
Rule 4: IF (CMF is VH and CSM is L) then Images are Partially Distinct
Rule 5: IF (CMF is VH and CSM is VL) then Images are Partially Distinct
Rule 6: IF (CMF is H and CSM is VH) then Images are Similar
Rule 7: IF (CMF is H and CSM is H) then Images are Similar
Rule 8: IF (CMF is H and CSM is M) then Images are Partially Similar
Rule 9: IF (CMF is H and CSM is L) then Images are Partially Distinct
Rule 10: IF (CMF is H and CSM is VL) then Images are Partially Distinct
Rule 11: IF (CMF is M and CSM is VH) then Images are Similar

<table>
<thead>
<tr>
<th>CMF/CSM</th>
<th>Very High</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
<th>Very Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very High</td>
<td>VH-VH</td>
<td>VH-H</td>
<td>VH-M</td>
<td>VH-L</td>
<td>VH-VL</td>
</tr>
<tr>
<td>High</td>
<td>H-VH</td>
<td>H-H</td>
<td>H-M</td>
<td>H-L</td>
<td>H-VL</td>
</tr>
<tr>
<td>Medium</td>
<td>M-VH</td>
<td>M-H</td>
<td>M-M</td>
<td>M-L</td>
<td>M-VL</td>
</tr>
<tr>
<td>Low</td>
<td>L-VH</td>
<td>L-H</td>
<td>L-M</td>
<td>L-L</td>
<td>L-VL</td>
</tr>
<tr>
<td>Very Low</td>
<td>VL-VH</td>
<td>VL-H</td>
<td>VL-M</td>
<td>VL-L</td>
<td>VL-VL</td>
</tr>
</tbody>
</table>
Rule 12: IF (CMF is M and CSM is H) then Images are Partially Similar
Rule 13: IF (CMF is M and CSM is M) then Images are Partially Similar
Rule 14: IF (CMF is M and CSM is L) then Images are Partially Distinct
Rule 15: IF (CMF is M and CSM is VL) then Images are Partially Distinct
Rule 16: IF (CMF is L and CSM is VH) then Images are Partially Similar
Rule 17: IF (CMF is L and CSM is H) then Images are Partially Similar
Rule 18: IF (CMF is L and CSM is M) then Images are Partially Distinct
Rule 19: IF (CMF is L and CSM is L) then Images are Partially Distinct
Rule 20: IF (CMF is L and CSM is VL) then Images are Partially Distinct
Rule 21: IF (CMF is VL and CSM is VH) then Images are Partially Distinct
Rule 22: IF (CMF is VL and CSM is H) then Images are Partially Distinct
Rule 23: IF (CMF is VL and CSM is M) then Images are Partially Distinct
Rule 24: IF (CMF is VL and CSM is L) then Images are Partially Distinct
Rule 25: IF (CMF is VL and CSM is VL) then Images are Distinct
4.3 Results and Evaluation
This section documents the SEBI experiment results. In section 4.3.1, we measure the precision, recall, f-measure and query execution time of SEBI framework by utilizing different similarity measurement algorithms.

### 4.3.1 SEBI Results and Evaluation

The SEBI overall experiment results and evaluation have been divided into two phases. In the first phase, the precision, recall, f-measure and query execution time (QET) of SEBI are computed by applying the vector space model (cosine similarity) [172] and fuzzy C-Means [175] on the SEBI semantic annotations. In the second phase, popular similarity measurement algorithms such as Jaccard coefficient, Euclidean distance and Dice coefficient along with fuzzy C-means are employed separately into SEBI and the precision, recall, f-measure [176] and query execution time (QET) are recorded. The purpose of computing the similarity with the multiple algorithms is to observe the performance of each algorithm to determine the most suitable algorithm on RDF graph data generated by SEBI and stored in iCyrus [24].

Precision, Recall and F-measure [176] are well-known reliable measures to evaluate the performance of an information retrieval system. Precision or positive predictive value is the fraction of retrieved instances that are relevant. However, recall or sensitivity is the fraction of relevant instances that are retrieved. F-measure is the harmonic mean of both precision and recall, and it provides an average value of system performance. Generally, a precision is considered a qualitative measure while recall is computed to observe the quantitative values that have been retrieved in response to a user query from an information system or search engine. In our context, a high precision shows that the system has returned more relevant sequence images than irrelevant. However, the high recall shows that the system has fetched mostly relevant sequence images. The
SEBI similar image finding program permits a user to include or exclude parameters about which a user wants to calculate the similarity. Following are the parameters that could potentially be used in the similarity measuring criteria: hasBLASTDescription, hasSimilarPrositeSignature, hasSameMOTIF, hasSameDisease, pHmmerScanKingdom, hasSameClan, hasSameChebiDictTerms, hasHMMERTarget, hasSamePMCSpecies, hasSameGoTerms, hasSimilarPhmmerTarget, hasPFamily, hasSameArticle, hasImageType, hasAltName, hasBLASTAccession, hasGeneType and hasScanPrositePrecursor.

A user can set any number of parameters as input in the similarity measurement algorithm. Since the similarity measurement relies on the provided parameters, the computation time also varies with the change in parameter numbers. The extracted images are categorized from exactly identical to totally irrelevant or distinct images. To evaluate the information retrieval performance of SEBI, we grouped queries into four batches prior to executing the experiments in order to observe performance batch-wise. Table 4.3 shows the precision, recall, f-measure and query execution time batch-wise by using the cosine similarity. Formulae to calculate the precision, recall and F-measure are defined below. Here tp and fp are true positive, fp is the false positive and fn is the false negative.

\[
\text{Precision} = \frac{tp}{tp + fp} \quad (5)
\]

\[
\text{Recall} = \frac{tp}{tp + fn} \quad (6)
\]

\[
F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (7)
\]
Table 4.3: The Overall SEBI Performance Evaluation

Table 4.3 shows the overall performance of the SEBI system. Cosine similarity algorithm has been applied on the graph data and similarity score has been calculated. SEBI generates the 0.94 ~ 0.96 F-measure with cosine similarity algorithm.

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>QET Batch-wise (In second)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch 1</td>
<td>0.95</td>
<td>0.92</td>
<td>0.94</td>
<td>398</td>
</tr>
<tr>
<td>Batch 2</td>
<td>0.97</td>
<td>0.86</td>
<td>0.91</td>
<td>817</td>
</tr>
<tr>
<td>Batch 3</td>
<td>0.93</td>
<td>0.97</td>
<td>0.95</td>
<td>691</td>
</tr>
<tr>
<td>Batch 4</td>
<td>0.93</td>
<td>0.98</td>
<td>0.96</td>
<td>733</td>
</tr>
</tbody>
</table>

Table 4.4 displays a performance comparison matrix of SEBI with different similarity measurement algorithms. It is important to note that the Dice and Euclidean measures do not handle blank and redundant nodes; therefore, precision does not show a satisfactory result. Jaccard comes with no normalization of unit hyper-sphere; therefore, it produces low precision, recall and F-measure compared with other algorithms. The results in Table 4.4 show that the vector space model (Cosine similarity) generates more favorable results than Jaccard coefficient, Euclidean distance and Dice coefficient.
Table 4.4: SEBI Performance Evaluation

Table 4.4 displays the statistics that have been recorded by employing the Jaccard, Euclidean, Dice and Cosine similarity algorithms. The purpose was to apply the popular similarity calculation algorithm within SEBI and recommend the most suited algorithm.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>QET (In second)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Batch 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jaccard Coefficient</td>
<td>0.65</td>
<td>0.87</td>
<td>0.75</td>
<td>153</td>
</tr>
<tr>
<td>Euclidean Distance</td>
<td>0.54</td>
<td>0.69</td>
<td>0.61</td>
<td>123</td>
</tr>
<tr>
<td>Dice Coefficient</td>
<td>0.67</td>
<td>0.78</td>
<td>0.73</td>
<td>223</td>
</tr>
<tr>
<td>Cosine Similarity</td>
<td>0.95</td>
<td>0.92</td>
<td>0.94</td>
<td>398</td>
</tr>
<tr>
<td><strong>Batch 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jaccard Coefficient</td>
<td>0.43</td>
<td>0.48</td>
<td>0.46</td>
<td>779</td>
</tr>
<tr>
<td>Euclidean Distance</td>
<td>0.58</td>
<td>0.69</td>
<td>0.64</td>
<td>633</td>
</tr>
<tr>
<td>Dice Coefficient</td>
<td>0.76</td>
<td>0.63</td>
<td>0.69</td>
<td>695</td>
</tr>
<tr>
<td>Cosine Similarity</td>
<td>0.97</td>
<td>0.86</td>
<td>0.91</td>
<td>817</td>
</tr>
<tr>
<td><strong>Batch 3</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jaccard Coefficient</td>
<td>0.47</td>
<td>0.78</td>
<td>0.58</td>
<td>599</td>
</tr>
<tr>
<td>Euclidean Distance</td>
<td>0.61</td>
<td>0.69</td>
<td>0.65</td>
<td>619</td>
</tr>
<tr>
<td>Dice Coefficient</td>
<td>0.58</td>
<td>0.63</td>
<td>0.61</td>
<td>648</td>
</tr>
<tr>
<td>Cosine Similarity</td>
<td>0.93</td>
<td>0.97</td>
<td>0.95</td>
<td>691</td>
</tr>
<tr>
<td><strong>Batch 4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jaccard Coefficient</td>
<td>0.55</td>
<td>0.48</td>
<td>0.52</td>
<td>645</td>
</tr>
<tr>
<td>Euclidean Distance</td>
<td>0.49</td>
<td>0.69</td>
<td>0.57</td>
<td>563</td>
</tr>
<tr>
<td>Dice Coefficient</td>
<td>0.74</td>
<td>0.63</td>
<td>0.68</td>
<td>400</td>
</tr>
<tr>
<td>Cosine Similarity</td>
<td>0.93</td>
<td>0.98</td>
<td>0.96</td>
<td>733</td>
</tr>
</tbody>
</table>
Figure 4.4 displays the Precision, Recall, F-Measure and total query computation time (QET) with Jaccard coefficient, Euclidean distance, Dice coefficient and cosine similarity.

![Figure 4.4: A Graphical Comparison Among Similarity Measures Using the SEBI Data](image)

[Figure 4.4 shows the average recall, precision and F-measure for Jaccard, dice, Euclidean and Cosine similarity cases. It’s obvious from the graph that the cosine similarity performs better than other algorithms in SEBI environment.]

Similarly, Figure 4.5 depicts the query execution time comparison among popular similarity calculation methods for SEBI. Here Jaccard and cosine similarity algorithms take slightly more time than the other two. Jaccard takes more time because of its algorithmic complexity; however, the cosine similarity [172] algorithm is more rigorous in the normalization to increase the precision rate. Jaccard is algorithmically more complex than Dice because Jaccard does not provide any criteria to normalize into unit hyper sphere. For simplicity, assume there are three images with annotations among which we have to calculate the similarity score using Jaccard coefficient. If any image comes with duplicated annotations, Jaccard counts the redundant annotations and consequently the algorithm has to compute for the same annotation multiple times instead of
considering the annotation once. Therefore, we can say that the lack of a normalization function by default makes the algorithm computationally complex.

![Figure 4.5: Queries Computation Time Comparison Batch wise](image.png)

*Figure 4.5: Queries Computation Time Comparison Batch wise*

*Figure 4.5 compares the query execution time by applying the Jaccard, dice, Euclidean and cosine similarity one by one in SEBI system. The graph shows that overall query execution takes more time in case of Jaccard and cosine in comparing with Dice and Euclidean. Jaccard by default cannot normalize the repeated term and has to calculate the similarity’s score for all the values.*

4.4 Discussion

This chapter illustrates the assessment we conducted to measure the sequence image similarity based on their semantic annotations and usability of the system to retrieve relevant sequence images. To measure the semantic similarity among sequence images with annotations, this chapter contributes by applying the vector space model on image graph data (RDF image data). This model diminishes the effect of redundant RDF nodes and increases the weight of RDF nodes that occur rarely to generate a normalized semantic similarity score. Moreover, the adoption of Fuzzy C-Means [175] for categorization of sequence images makes it possible that the sequence images
could be categorized based on their degree of membership to certain categories or clusters although it was not evaluated objectively.

SEBI generates the annotation through SADI web services [95] and stores them in a triple store [24]. The sequence similarity algorithm in SEBI takes the generated annotation as input and calculates the similarity among sequence images. To examine the performance of SEBI, we executed the use cases queries in two phases and recorded the precision, recall, f-measure and query execution time. In the first phase, all the experiments were conducted by employing the cosine similarity to measure the sequence image similarity and fuzzy c-means for image categorization. In the second phase, we recruited the Jaccard, Euclidean and Dice Coefficient similarity algorithms and applied the same queries under the same experiments setup. Our experiments exhibit that by using the cosine similarity we can achieve up to 94% precision based on a corpus.

However, as displayed in Figure 4.5, the overall time for query execution in case of cosine similarity is more than the time needed to execute the same queries under Jaccard, Dice and Euclidean algorithms. In the cosine similarity, cosine of the angle is measured among the query vectors and all the candidate vectors after the frequency and length normalization. The ranking criteria also take some additional time. In SEBI, query execution time is highly dependent on hardware specification of the system; especially, the RAM or memory plays a significant role in run-time computation. By employing the distributed computation techniques, we can decrease the on-the-fly computation time. In chapter 5, we discuss a few possible techniques to enhance the overall performance of the SEBI system. One of the advantages of a semantic web based system such as SEBI is that it permits dynamic integration of sequence images with third party systems.
As SEBI provides the sequence images and their related images, it can be integrated into any system where the users like to extract scientific knowledge in quasi-real-time.

In contrast with SEBI the available image retrieval solutions retrieve the similar image based on their actual contents. A number of CBIR (content-based image retrieval) techniques have been introduced in [166, 177]. CBIR techniques incorporate the visual feature (geometry of an object) extraction algorithms to extract image features such as color, texture and shape prior to apply the indexing to make the images discoverable. Several commercial software and prototypes for image similarity indexing have been introduced such as Photobook [167], Virage [178], VisualSEEK [179], Netra [180], SIMPLIcity [181] in the last decade and they all use CBIR based techniques for image retrieval. PhenoImageShare [182] is another Solr powered ontology-enabled solution to annotate and discover the phenotypic images. Unlike SEBI, PhenoImageShare does not employ any web service or automatic annotation generation method rather it provides an environment where users annotate the phenotypic images manually. Another difference of SEBI with PhenoImageShare is that SEBI works on sequence image data while PhenoImageShare is restricted to phenotypic images.
Chapter 5

Conclusion

This thesis introduces the SEBI system and framework for the semantic enrichment and similarity approximation of biomedical sequence images published in scientific literature. Semantic enrichment of biomedical sequence images (SEBI) is a fundamental step towards an image-first knowledge discovery paradigm [5]. The core of the SEBI framework is the authoring of sequence image annotations that describe specific features of a sequence image’s content. These annotations are generated using a semantic enrichment process (section 1.3) following a series of image pre-processing (section 3.2.1) and sequence image segregation (section 3.2.6) steps, including optical character recognition (section 3.2.2) of biological sequence information in sequence images. Annotations are generated by invoking web services and published as RDF graphs (sections 4.2.1 and 4.2.2) which are persisted as triples in a triple store. The resulting semantically enriched
images are readily reusable and can be recruited in ad-hoc image data integration projects. To augment the discovery of biomedical sequence images we propose the use of an algorithm to compute image similarity based on semantic annotations (section 4.2.3).

In addition to proposing a novel use of semantic annotations permitting image discovery and reuse, this research work seeks to lay a foundation for a new paradigm in information retrieval and knowledge discovery whereby targeted access to scientific knowledge is mediated primarily through image search, discovery of related images (see section 4.1.1) and linking to source publications describing scientific investigations. In this paradigm, instead of seeking scientific literature by entering keywords, users will first query for an image, confirm its relevance to their goals and, based on its semantic similarity to other images, find other source publications where the related image was first published. This is called image-first search [5]. This thesis also details an evaluation of performance where a number metrics have been employed both at component and system levels (section 3.2.6, 4.3.1). The performance evaluation process has rendered favorable results that demonstrate the legitimacy of key components of the proposed solution. This approach to the retrieval of semantically similar sequence images may in future provide a valuable alternative or addition to current best practices in knowledge discovery.

5.1 Thesis Contributions

In this section, we highlight two main contributions that we have made in this thesis. The subsection 5.1.1 provides details of the contribution at the sequence image processing level, while 5.1.2 highlights the contribution at the sequence image similarity and categorization stage of the SEBI framework.
5.1.1 Sequence Image Processing

SEBI unlocks sequence image information to yield new annotations on sequence images in the second Tier of SEBI framework (See Figure 1.2). This section highlights our contribution at the sequence image processing stage, which involves sequence image acquisition and optical character detection and extraction as sub-stages. For the image acquisition and preprocessing stages, three algorithms have been developed, namely IAA (image acquisition algorithm), DIA (deep image analysis) and SISA (sequence image segregation algorithm). The heterogeneous image dataset that was obtained from the Yale Image Finder contained images with high and low quality resolution. Moreover, some images were redundant and contained incomplete metadata. In YIF images are persisted mainly in web-friendly formats such as Gif and Jpeg. Web-friendly image formats were not considered suitable for the OCR process and were not used, since image resolution should be at least 300×300 pixels in order to generate >70% level of accuracy from the optical character recognition process. To secure a usable dataset, the IAA establishes a concurrent connection with PubMed API and YIF to extract the incomplete/missing metadata and to detect and remove the redundant images. A crosscheck procedure of IAA removes the image duplication. To improve image quality, the IME (image manipulation engine) incorporates a number of image enhancement filters such as Gaussian and Laplace (see section 3.2.1) to increase contrast, resolution, dynamic range and color scheme of the acquired images.

The IME employs the ImageJ library [156], which can automatically detect the low-resolution images. Subsequently, IME invokes the IAA algorithm to find any higher resolution (see Figure 7.8) images from PubMed using image signatures (Tables 3.5 and 3.6). DIA has been developed to segregate sequence images from other image types i.e radiographs. SISA further separates sequence images into protein and DNA sequence image categories [18]. The process of extracting
accurate sequence ASCII (optical characters) from a sequence image is fraught with difficulties, and without domain-specific training of an OCR algorithm one cannot achieve the desired OCR accuracy. SEBI requires a comprehensive training and testing of its OCR algorithm [183] to achieve the maximum accuracy level and to reduce the algorithm response time. Therefore, to detect and extract the optical characters from sequence images we trained our algorithm with font specific (i.e. courier and times new roman) seed files along with Tesseract API 3.0 [184]. For the training process, 540 images were selected semi-automatically. Of these, 250 images were of low resolution and 150 images were of medium to high resolution. Some images contained fragmented sequence strings or contained sequences positioned at different parts of the image. We included 140 of these kinds of image in the training process. Tesseract 3.0, Leptonica image processing and ImageJ libraries along with jTessBoxEditor GUI [152] trainer were used to train the OCR algorithm. We trained the OCR algorithm with courier and time new roman font sets and removed the other font families from the Tesseract resource library primarily because sequence images mainly present their characters in these formats. Moreover, we disabled the dictionary feature in Tesseract 3.0 as we were not looking for any dictionary words from the sequence ASCII at the time of optical character extractions. To observe the accuracy of the algorithms, we introduced the character level accuracy scale [28], which provides a summarized overview of the accuracy of any OCR system. For the testing process, we used 200 randomly selected sequence images with different resolution and applied the OCR algorithms. Overall our algorithm yielded an average character level accuracy between 69.5% ~ 98.6%. We noticed that image preprocessing, font-specific training, disabling of the dictionary and pre-setting the image layout option increased the OCR accuracy and reduced the optical character detection and extraction time.
5.1.2 Sequence Image Similarity and Categorization

SEBI measures the sequence image similarity based on derived annotations. This section explains our contribution at the semantic enrichment, biomedical image similarity and categorization stages.

In this thesis, we introduced semantic image enrichment, which is a technique to annotate sequence images with the relevant semantic annotations describing their contents. To generate annotations on sequence images we proposed the use of SADI semantic web services that can provide interoperability of BioNLP and sequence analysis services and the ad-hoc consolidation of their results. We exposed the sequence analysis tools such as BLAST, HMMER and PROSITE (see section 2.5) as web services through the SADI framework [95] to yield annotations from optical characters strings derived from sequence images. Moreover, to extract the drug, chemicals, Go-terms, protein, and species from the image-associated text we developed relevant web services by exposing the available NLP software such as Whatizit [96]. A total of thirty web services have been developed which can be accessed from the SEBI web service registry shown in Table 4.1. To support the management of SEBI generated annotation, the BIM ontology was designed to provide the necessary semantic vocabularies to convert the image annotations into RDF. A LDM (linked data manager) was introduced in the SEBI framework (See Figure 1.2 Tier 3), which leverages the BIM ontology to convert and store the annotations into RDF. LDM stores the annotations into triple-store iCyrus [24]; which was configured according to linked data specifications such that it behaves as an interoperable API for biomedical images. iCyrus is powered by a dedicated semantic architecture that exposes the YIF contents as linked data, permitting integration with related information resources and consumption by linked data-aware data services.
Since sequence images are often identical in appearance, one possible way to measure the similarity between the sequence images is to calculate the semantic similarity based on their annotations. The SEBI framework takes sequence image annotations from iCyrus and computes the similarity among the images (as displayed in Figure 9.1). In SEBI, we employed the cosine similarity which is based on Vector Space Model (VSM) [21] to measure the similarity among sequence images’ annotations. VSM is essentially an algebraic model that represents textual information or documents in a corpus as a point(s) or vector(s) in a vector space. Points or vectors that appear closer are considered related to each other, and points that appear far apart are regarded as distant [167]. Our strategy preserves the semantics of the annotations while applying the VSM to calculate the similarity. To classify the biomedical sequence images, cosine similarity was paired with Fuzzy C-Means clustering in this thesis. Fuzzy C-Means [168] is a powerful clustering technique that permits one piece of data to belong to multiple clusters or categorizes. To examine the efficiency of SEBI, we performed experiments employing the cosine similarity and Jaccard, Euclidean, Dice Coefficient similarity algorithms for cross comparison of SEBI performance. Our experiments demonstrate that by using the cosine similarity we can achieve a precision of up to 94%; in contrast other similarity measurement techniques we have recorded a relatively low precision rate [167].
5.2 Thesis Limitations

One of the major limitations that we faced within the SEBI system is the slowness of the searching algorithm that seeks to find the related sets of images based on their semantic annotations. The underlying algorithm takes the graph data (rdfized image dataset) and plots the vectors (see section 4.2.3) representing each image-graph virtually in a unit hyper sphere. The cosine similarity algorithm measures the cosine of an angle between each vector (representing an image annotation) and further calculates the corresponding similarity score between the sequence images. The cosine similarity algorithm also monitors for term repetition, which in the case of a graph is the node frequency, and normalizes this prior to calculating the cosine of angle among vectors. It is important to note that the overall query execution time depends upon the number of annotations involved in search criteria defined by the user. For instance, in a case where there are one hundred sequence images in a triplestore along with their semantic annotations, and each image has twelve annotations and the search query has ten parameters in it to match, then the algorithm has to calculate the similarity score roughly twelve thousand times. With a desktop PC (4 GB RAM and i5 processor), it may take a few minutes to display the results. Limiting the query parameters while searching can increase the query response time. However, decreasing the number of annotations in the search criteria directly affects the overall precision of the search results.

The sequence image similarity criteria in SEBI are limited to the directly derived annotations. To improve the precision rate additional annotations could be included. For example, adding upper-level hierarchical information about directly retrieved annotations from controlled vocabulary such as the Gene Ontology could be included. Additional weights could be applied in the vector space model to account for the distance from hierarchical terms to the core annotations.
Another set of annotations that could be included in image similarity computation are those derived from the text associated with an image, where images are from publications. Within the current setup, BioNLP algorithms in SEBI detect and extract the biologically relevant information from the metadata (e.g. captions, descriptions) of the images and normalize the entities to the canonical names that have been defined in online resources e.g. GO, DrugBank.

Similarly, the sequence analysis services extract information about a protein clan and this associated information such as clan members, possible alignment, interactions and family relationships can be extracted from http://pfam.xfam.org/. The taxonomy map available at http://www.uniprot.org/taxonomy/ also provides valuable information that can be used to extend the SEBI search criteria.

Within the current SEBI framework, the semantic enrichment process is performed only on the sequence images. Although other images have been added along with default annotations such as caption, title, abstract, PMID, PMCID and DOI into iCyrus [24], no new annotations have been generated for those images. Images such as pathway and GEL images can be annotated from the available related resources such as KEGG, but were not investigated in this thesis.

Since annotations are limited to those represented in the BIM ontology further extension of this conceptualization is appropriate for scaling SEBI, specifically the classes and property structure should be extended by adding additional domain specific axioms and rules into the ontology. A final consideration to the performance of SEBI is memory allocation. This is both a hardware and software level limitation. To overcome the data loss issue due to system failure, we limited the batch processing time for each experiment. For instance, the maximum query timeout limit (per set) was set to 13 minutes for simple and complex queries and 30 minutes for mixed SPARQL queries with large size data. One reason to set a time limit for queries was to protect
against information loss in case of memory overload and other hardware failure since data was
distributed on different endpoints that resided physically on different machines. It might be
possible that certain queries stopped responding because of a shortage of system memory and other
hardware failure. Commercial clustering solutions provided by Amazon and Google such as
Amazon EC2 and Google cloud could be possible solutions for the lengthy complex search query
execution time. In summary, although this thesis introduces a novel approach to search for similar
images, the scientific contribution of this work is limited to the discovery of published biomedical
sequence images. We have not evaluated the performance of the system or applied the SEBI
framework on general image types outside the biomedical image domain.

5.3 Future Work

There could be several options through which the SEBI system can be extended in future to
improve the system efficacy. One possible way is to configure the system in a concurrent
computing environment [185]. This is an approach where multiple computational requests are
carried out in a concurrent computing setup. The basic principle is to divide a big problem into
small tasks and different processors on the same machine then solve each task at the same time. In
parallel computing [186], memory is shared among different processors, whereas in distributed
computing [187], each processor has its own memory allocated on the small tasks and the results
are integrated at the end of computation. In SEBI, where the use cases require computation of a
similarity score based on multiple semantic similarity criteria, the algorithm can be configured to
work in concurrent computing environment. Available distributed computing platforms such as S4
[188] and SPC [189] support a developer to rapidly develop a new application or configure an
existing one according to the distributed computing setup.
S4, which is the abbreviation of Simple Scalable Streaming System, is a scalable distributed processing engine [188] that could a candidate for future performance enhancement. A distributed computing platform has been used to enhance image processing and real-time indexing activities in [190].

MapReduce is a model that inspires the working model of S4, and has been designed specifically to fulfill the requirements of real-time data mining-based information search engines. The S4 platform architecture is compatible both with the research and production environments. Research projects require an environment where a new algorithm could be deployed in a short time, whereas in a production environment, scalability is considered a major concern [189]. Likewise, the Stream Processing Core (SPC) is a distributed computing platform that computes the information while retrieving data from various digital data streams.

Adoption of CUDA [191] in SEBI, which leverages the computing power of graphic processing unit would increase the processing speed of DIA and SISA algorithms. However, a developer would have to optimize the algorithm code according to the CUDA specifications. As explained in Chapter 4, SEBI generates annotation through multiple sequence analysis and BioNLP web services. This continuously generated annotation is considered as a stream of data, and searching can be managed by employing the methods described in [186, 189]. As detailed in Chapter 4, SEBI employs the cosine similarity to extract the related sequence images. The overall searching criteria can be optimized by adopting heuristic algorithms such as Genetic Algorithm (GAs) [192] in combination with a cosine similarity technique. In artificial intelligence (AI), GA is considered to be part of evolutionary computational algorithms, and it has been designed to mimic the natural process of evolution to discover solutions to problems [193]. GA resolves the search optimization issues in the information retrieval domain effectively because
of its robust and flexible underlying mechanism. Another, future work could be the development of the live version of SEBI, where the algorithms will check for the new articles published online periodically and will pull the newly added papers to extract their images, performing the semantic and similarity approximation process on the fly. Available triple stores such as Halyard [194] which provides support for horizontal scalability, Named Graphs and querying with SPARQL 1.1, and BlazeGraph [195] which is a GPU accelerated high performance triple store for large RDF graphs could make it possible to store SEBI run-time generated annotations and perform fast queries and subsequently faster sequence image similarity comparisons.

Overall, the key limitation of SEBI in its current implementation is the resource- (CPU and RAM) hungry nature of the similarity computation algorithms. Computation time is highly dependent on hardware. To date, semantic annotation of images as described in this thesis has been successful and has laid the groundwork for developing image search techniques based on image similarity. Based on this, new navigation tools for knowledge discovery can be designed that may well rival traditional literature navigation techniques. If fully deployed, such systems have the potential to significantly change the behaviors of many scientists and how libraries archive literature.
Appendix -1

Glossary

**Ad-hoc RDF Integration** Two or more RDF graphs can be integrated on the fly if they share same URIs scheme.

**BioNLP Pipeline** a BioNLP pipeline consists of one or more biological natural language processing algorithms, which often run serially to extract the biological content from a document.

**Biological component** refers to the unique, highly organized substances of which cells, and thus living organisms, are composed. For more information, please read [http://bioserv.fiu.edu/~walterm/FallSpring/review1_fall05_chap_cell3.htm](http://bioserv.fiu.edu/~walterm/FallSpring/review1_fall05_chap_cell3.htm).

**Drug Bank** The DrugBank database is a comprehensive, high quality, freely accessible, online database containing information on drugs and drug targets.

**Fuzzification** The fuzzification comprises the process of transforming crisp values into grades of membership for linguistic terms of fuzzy sets. The membership function is used to associate a grade to each linguistic term.

**FASTA** format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences. The format originates from the FASTA software package, but has now become a standard in the field of bioinformatics.

**Fuzzy Membership** The membership function of a fuzzy set is a generalization of the indicator function in classical sets. In fuzzy logic, it represents the degree of truth as an extension of valuation.
**Linguistic Variables** A linguistic variable such as age may have a value such as young or its antonym old. However, the great utility of linguistic variables is that they can be modified via linguistic hedges applied to primary terms. The linguistic hedges can be associated with certain functions.

**Federated Querying** Federated query is the ability to take a query and provide solutions based on information from many different sources.

**Grounding (Mutation)** Extract relations between mutations mentioned in image captions and proteins mentioned in full text is called Mutation Grounding.

**Linked Data** describes a method of publishing structured data so that it can be interlinked and become more useful.

**Mutation** In genetics, a mutation is a change of the nucleotide sequence of the genome of an organism, virus, or extra chromosomal genetic element.

**Protein–DNA interaction** is when a protein binds a molecule of DNA, often to regulate the biological function of DNA, usually the expression of a gene.

**Quality (Image)** is a composite term, which is based on parameters such as contrast, resolution, noise, dynamic range and color scheme etc. Image quality is used to measures the perceived image degradation.

**SPARQL** is an RDF query language, that is, a query language for databases, able to retrieve and manipulate data stored in Resource Description Framework format.

**Semantic enrichment** A method in which annotations describe as an RDF format. RDF is a data structure, which arranges the concepts into subject-predicate-object shape. To read more about RDF [http://www.w3.org/RDF/](http://www.w3.org/RDF/).
**SHARE** is an experimental federated query client featuring automatic discovery and orchestration of SADI services.

**Resolution (Image)** is one the image quality assessment parameters and refers to the number of pixels in an image. Resolution is sometimes identified by the width and height of the image as well as the total number of pixels in the image.

**REST** (REpresentational State Transfer) is an architectural style, and an approach to communications that is often used in the development of Web services.

**UniProt** is a comprehensive, high quality and freely accessible database of protein sequence and functional information, many entries being derived from genome sequencing projects. It contains a large amount of information about the biological function of proteins derived from the research literature.

**URIs** in computing, a uniform resource identifier (URI) is a string of characters used to identify a name of a web resource. Such identification enables interaction with representations of the web resource over a network, typically the World Wide Web, using specific protocols.

**WSDL** is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information.


Pande, S., et al. EMBRACE-SysC for analysis of NoC-based spiking neural network architectures. in International Symposium on System-on-Chip. 2010. IEEE.


Curriculum Vitae

Full name of the Candidate: Syed Ahmad Chan Bukhari

Universities attended (with dates and degrees obtained):

Master of Engineering, Gyeongsang National University, Korea 2012

Master of Information Technology, University of the Punjab, Pakistan 2008

Publications

14. Bukhari, S.A.C., et al., CEDAR.

Conference Presentations:

1- BioNLP-SADI. ..............................................DILS 2013, Montreal
2- HC2LOD. ..............................................DILS 2013, Montreal
3- SEBI ....................................................CSHALS 2014, Boston
4- BIM Ontology ........................................ ICBO 2015, Portugal
5- iCyrus .............................................. SWAT4LS 2015, Cambridge