UNIVERSITA' DEGLI STUDI DI PAVIA

FACOLTA' DI INGEGNERIA
DIPARTIMENTO DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

DOTTORATO DI RICERCA IN BIOINGEGNERIA E BIOINFORMATICA
XXV CICLO - 2012

A FLEXIBLE CBR SYSTEM FOR TRANSLATIONAL RESEARCH SUPPORTED BY MEDICAL TERMINOLOGIES AND SCIENTIFIC LITERATURE

PhD Thesis by

MATTEO GABETTA

Advisor:
Prof. Cristiana Larizza

PhD Program Chair:
Prof. Riccardo Bellazzi

This one goes out to the oneS I love

This one goes out to the oneS I left behind

Acknowledgments

I owe much gratitude to my supervisor Professor Cristiana Larizza, her inspiration, insight and feedback have been invaluable. I would like to thank also Professor Riccardo Bellazzi for his precious advices.

I gratefully acknowledge all the people at BMI lab, who I have enjoyed collaborating and working with throughout these years.

Abstract (Italiano)

L'attività di ricerca descritta in questa tesi è stata condotta nell'ambito del panorama emergente della ricerca traslazionale, volta ad integrare diversi tipi di ricerca (di base, orientata al paziente e basata sulla popolazione) provenienti da diverse discipline, con l'obiettivo a lungo termine di migliorare la salute pubblica. In particolare, l'obiettivo principale del presente lavoro riguarda lo studio di tecniche di gestione della conoscenza e lo sviluppo di soluzioni software per supportare lo scambio di informazioni tra i diversi attori che operano nel contesto della ricerca traslazionale. Al centro di questa attività di ricerca sono state poste la gestione e la valorizzazione dei dati provenienti sia dalla ricerca di base, quali la quantità impressionante di letteratura scientifica disponibile, sia da quella orientata al paziente, come ad esempio il volume ancor più grande di dati raccolti durante il processo di cura dei pazienti dagli operatori sanitari e dai ricercatori.

La ricerca è stata condotta in due direzioni principali che hanno dato origine allo sviluppo di altrettanti sistemi: l'estrazione e la corretta gestione delle entità biomediche che sono contenute nella letteratura scientifica e la messa a punto di una infrastruttura di ragionamento che valuti il grado di similitudine di casi medici (cioè pazienti) sulla base delle loro caratteristiche.

Entrambi questi aspetti sono caratterizzati da una notevole eterogeneità dei dati: la letteratura scientifica contiene molti tipi di entità rilevanti per la ricerca biomedica (ad esempio geni, proteine, farmaci, malattie), mentre ogni organizzazione sanitaria tipicamente gestisce in autonomia i dati dei propri pazienti con i sistemi di codifica che meglio si adattano ai propri scopi specifici. Per ovviare a questo problema e per sfruttare tutte le potenzialità dei dati disponibili, il filo conduttore di questo lavoro è stato la flessibilità. Il sistema di *mining* della letteratura (*Literature Mining system*) è stato progettato per essere abbastanza flessibile per estrarre e gestire correttamente molte tipologie di entità biomediche e, inoltre, la sua architettura permette di estendere facilmente questa analisi a qualsiasi altro tipo di concetto. Dall'altro lato, il sistema di ragionamento basato su casi (*Case Based Reasoning system*) dimostra la sua flessibilità consentendo di raccogliere e confrontare casi di pazienti provenienti da fonti diverse evitando di imporre un modello fisso ai loro insiemi di attributi.

Il sistema di *mining* della letteratura ha il compito di gestire l'enorme quantità di dati provenienti dalla letteratura scientifica. Esso svolge diverse

funzioni: dal recupero dei dati grezzi (cioè gli articoli) da PubMed, all'effettivo processo di estrazione delle entità biomediche rilevanti dagli *abstract* e la successiva memorizzazione delle informazioni ottenute in un sistema di persistenza sufficientemente flessibile per gestire le diverse entità estratte. Inoltre, è stato sviluppato un sistema di *Literature Based Discovery* che sfrutta i dati estratti dalla letteratura per evidenziare le relazioni nascoste che potrebbero portare alla scoperta di nuova conoscenza.

Il sistema di *mining* della letteratura può essere utilizzato per scopi diversi: per riassumere una determinata serie di articoli con le entità di interesse che sono contenuti nei loro *abstract*, per recuperare un set di articoli in base ai concetti che sono citati nei loro *abstract* e per la scoperta di possibile nuova conoscenza attraverso la valutazione delle associazioni indirette presenti in letteratura.

Il sistema di ragionamento basato su casi esegue diverse operazioni al fine di confrontare casi medici, facendo fronte all'eterogeneità che li caratterizza: per prima cosa mappa i dati dei pazienti in arrivo sfruttando una rappresentazione comune basata sullo *Unified Medical Language System* (UMLS) e utilizza un algoritmo per il calcolo della distanza tra essi al fine di rappresentare questi casi in uno spazio in cui possa essere valutato il loro grado di vicinanza relativa. L'algoritmo sviluppato produce un punteggio di distanza semantica tra due casi rappresentati da una serie di attributi variabili. Un altro compito realizzato dal sistema di ragionamento basato su casi è la generazione di una *query* che possa essere inviata al sistema di *mining* della letteratura, o direttamente a PubMed, sulla base del set di attributi che caratterizzano un paziente, consentendo una interrogazione della letteratura orientata al paziente.

Il sistema di ragionamento basato su casi può essere utilizzato sia per identificare, dato un nuovo caso, il sottoinsieme di casi noti che risultano essere più simili ad esso secondo l'algoritmo di distanza adottato, sia per ottenere la matrice di distanza relativa ad un dato insieme di casi al fine di identificare potenziali *cluster* sulla base delle caratteristiche descrittive di ciascun caso.

Infine è stata sviluppata un'interfaccia grafica (*Graphical User Interface - GUI*) per entrambi i sistemi; questo strumento, che per il momento è in grado di accedere ad un insieme limitato di funzionalità dei sistemi di *mining* della letteratura e di ragionamento basato su casi, permette già di navigare ed interrogare PubMed con *query* create grazie al supporto delle terminologie mediche (UMLS in particolare); permette, inoltre, di sfruttare l'algoritmo di calcolo della distanza sui casi noti e, ancora, di collegare il dominio dei pazienti con quello della letteratura grazie all'interrogazione della letteratura orientata al paziente.

La valutazione delle tecniche sviluppate e dei sistemi realizzati è stata effettuata nell'ambito di due progetti di ricerca con caratteristiche fortemente traslazionali: il progetto INHERITANCE e il progetto ONCO-i2b2. INHERITANCE (*Integrated Heart Research In Translational Genetics of Cardiomyopathies in Europe*) è un progetto finanziato dalla

Commissione Europea che adotta un approccio multidisciplinare e multi-centro per studiare la genetica delle cardiomiopatie dilatative (*Dilated Cardiomyopthy - DCM*) ereditarie e per comprendere l'impatto e la gestione della condizione all'interno delle famiglie che soffrono di DCM. Il progetto è strutturato in sei aree di ricerca che studiano diversi aspetti della condizione della DCM e la strategia traslazionale si basa su un algoritmo clinico che cerca di determinare caratteristiche specifiche della malattia da associare ai diversi tipi di DCM o di suggerire specifici *pathway* genetici o metabolici della malattia.

ONCO-i2b2 è un'iniziativa informatica avviata dall'Università degli Studi di Pavia e dall'IRCCS Fondazione Salvatore Maugeri a Pavia per sostenere la ricerca clinica in oncologia; questo progetto si propone di sostenere la ricerca traslazionale in oncologia e sfrutta le soluzioni software implementate dal centro di ricerca "*Informatics for Integrating Biology and Bedside*" (i2b2) negli Stati Uniti.

Il sistema di *mining* della letteratura è stato valutato all'interno di INHERITANCE. In particolare, è stato utilizzato per estendere l'insieme di geni causanti la DCM con nuovi geni, non ancora presi in considerazione, sulla base delle entità biomediche estratte dalla letteratura specifica relativa ad ogni singolo gene. Inoltre, l'analisi della letteratura è stata sfruttata per assegnare una priorità alla serie di geni che causano la DCM a partire dalle caratteristiche di un singolo caso clinico o, in alternativa, con l'obiettivo di realizzare una prioritizzazione di tutti i geni per la malattia. Infine, anche il sistema di *Literature Based Discovery* è stato testato nell'ambito di questo progetto, con l'obiettivo di inferire l'associazione tra DCM e suoi geni causanti, sfruttando solo la letteratura antecedente le loro associazioni esplicite.

Il sistema di ragionamento basato su casi è stato testato su entrambi i progetti con obiettivi diversi: in INHERITANCE con l'obiettivo di selezionare, tra una set di pazienti simulati affetti da DCM (creati col supporto dei medici coinvolti nel progetto), quelli che dimostrano di avere un maggior grado di vicinanza, al fin di testare la capacità del sistema di CBR di selezionare pazienti con diagnosi simili sulla base delle loro caratteristiche. In ONCO-i2b2 l'obiettivo è stato quello di estendere gli strumenti di interrogazione del *data warehouse di i2b2* con l'algoritmo di calcolo della distanza, al fine di migliorare il processo di selezione dei pazienti. Lo scopo reale di questa valutazione è stato quello di mostrare come il sistema di ragionamento basato su casi possa essere integrato con un *data warehouse*, rendendo più flessibili i suoi strumenti di interrogazione e navigazione dei dati.

La valutazione effettuata ha mostrato risultati promettenti e, in particolare, è diventato chiaro come la flessibilità nella gestione dei dati dei sistemi realizzati possa essere sfruttata in diversi ambiti. Infatti, questi sistemi sono stati utilizzati per scopi differenti dimostrando un alto grado di adattabilità; inoltre, le tecniche adottate per il loro sviluppo e la loro effettiva implementazione garantiscono anche un elevato grado di estensibilità a diversi contesti applicativi. Ad esempio, il sistema di *mining*

della letteratura può essere integrato con nuovi strumenti per l'estrazione di informazioni, al fine di estrarre nuovi tipi di concetti, mentre la natura stessa del sistema di ragionamento basato su casi, con la sua rappresentazione dei casi basata su UMLS, permette di aggiungere all'ambiente di ragionamento nuovi casi provenienti da diverse fonti.

Le direzioni future della ricerca descritta in questa tesi sono molteplici. Esse comprendono, per esempio, l'estensione e l'affinamento delle tecniche di *text mining* impiegate nel sistema di *mining* della letteratura e la valutazione di nuovi algoritmi di distanza per permettere al sistema di considerare diversi tipi di attributi (ad esempio, misure cliniche e dati genomici). Inoltre, la valutazione del sistema di ragionamento basato su casi deve essere portata avanti, in particolare, per testare le prestazioni quando si confrontano pazienti caratterizzati da un numero di attributi che superi quello relativo ai casi analizzati fino ad oggi. Oltre allo sviluppo di nuove o più raffinate metodologie è conveniente dedicare parte dello sforzo per completare l'implementazione dell'interfaccia grafica del sistema al fine di integrare adeguatamente tutte le funzionalità dei sistemi sviluppati e ottenere uno strumento che possa essere effettivamente utilizzato nella pratica clinica e nella ricerca.

Abstract (English)

The research activity described in this thesis has been conducted within the emergent scenario of translational research, aimed at integrating different types of research (basic, patient-oriented and population-based research) from different disciplines with the long-term goal of improving the public health. In particular, the main target of the present work concerns the investigation of knowledge management techniques and the development of software solutions to support the information exchange between the several actors operating in the translational research context. The focus of this research activity has been set on the management and enhancement of the data coming both from basic research, such as the impressive amount of scientific literature available, and from patient-oriented research, such as the even larger volume of patients data collected by healthcare providers and researchers.

The research efforts have been divided in two main directions that have given rise to the development of as many systems: the extraction and proper management of the biomedical entities that are contained in the published literature and the set up of a reasoning infrastructure that assesses the degree of closeness of medical cases (i.e. patients) on the basis of their characterizing features.

Both these aspects are affected by a noticeable data heterogeneity: scientific literature contains many types of entities relevant for biomedical research (e.g. genes, proteins, drugs, diseases) while every healthcare organization typically manages its own patients' data autonomously according to the codification systems that better suite its specific purposes. In order to overcome this issue and, moreover, to exploit all the potentialities of the available data, the leading thread of this work has been flexibility. The Literature Mining system has been designed to be flexible enough to extract and properly manage many relevant biomedical entities and, moreover, its architecture allows to easily extend this analysis to any other concept type. On the other side, the Case Based Reasoning (CBR) system proves its flexibility by allowing to collect patient cases from disparate sources and by avoiding to force them to conform their feature set to a fixed model.

The Literature Mining system deals with the enormous amount of data coming from the published scientific literature. It accomplishes several tasks: from the retrieval of the raw data (i.e. the articles) from PubMed to the actual Information Extraction process of the relevant entities from their abstracts and the successive storage of the achieved information in a

persistence layer flexible enough to manage the many different entities extracted. Moreover, a Literature Based Discovery system has been developed in order to exploit the data extracted from the literature and point out hidden relations that could lead to the discovery of new knowledge.

The Literature Mining system can be used for different purposes: the summarization of a given set of articles with the entities of interest that are contained in their abstracts, the retrieval of a literature set according to the concepts that are cited into the articles' abstracts and the discovery of potential new knowledge through the evaluation of the indirect associations made in literature.

The CBR system performs several tasks to compare medical cases dealing with their heterogeneities: it maps the incoming patient data onto a common representation framework based on the Unified Medical Language System (UMLS) and exploits a distance score algorithm to represent these cases in a space where their relative degree of closeness can be evaluated. The developed algorithm produces a semantic distance score between two cases represented by a variable set of features. Another task accomplished by the CBR system is the generation of a query that can be submitted to the Literature Mining system, or directly to PubMed, on the basis of the feature set of a patient, thus allowing a patient-oriented literature interrogation.

The CBR system can be used both for identifying, given a new case, the subset of known cases that prove to be more similar to it according to the distance algorithm; and to achieve the distance matrix relative to a given set of cases in order to identify clusters on the basis of the descriptive features of each case.

Finally, a Graphical User Interface for both the systems has been developed; although this tool, for the moment, is able to access to a limited set of functionalities of the Literature Mining system and the CBR system, it already allows to navigate PubMed and interrogate it with *ad hoc* queries generated with the support of medical terminologies (UMLS in particular), to exploit the distance score algorithm on the available known cases and to link the patient-domain with the literature-domain with patient-orient PubMed queries.

The evaluation of the adopted techniques and the implemented systems has been performed within two research projects with strong translational traits: the INHERITANCE project and the ONCO-i2b2 project. INHERITANCE (Integrated Heart Research In Translational Genetics of Cardiomyopathies in Europe) is a multidisciplinary, multi-center research project funded by the European Commission that seeks to study the genetics of inherited Dilated Cardiomyopathy (DCM) and to understand the impact and management of the condition within families that suffer from DCMs. The project is structured into six research areas that study different facets of the DCM condition and its translational strategy is based on a clinical algorithm that seeks to determine disease-specific features to be associated with different types of DCM or suggest specific genetic or metabolic pathways of disease.

ONCO-i2b2 is an information technology initiative started by the University of Pavia (Italy) and the IRCCS Fondazione Salvatore Maugeri hospital in Pavia to support clinical research in oncology; this project aims at supporting translational research in oncology and exploits the software solutions implemented by the Informatics for Integrating Biology and the Bedside (i2b2) research center in the U.S.

The Literature Mining system has been evaluated within INHERITANCE; it has been used to extend the set of candidate DCM-causing genes with new, still-not-evaluated, genes on the basis of the entities extracted from each gene-specific literature. Moreover, the performed literature analysis has been exploited to prioritize the set of DCM-causing genes starting from a single case or, in alternative, with the goal of achieving a general prioritization list for the disease. Finally, the Literature Based Discovery system has been tested within this project, by demonstrating that it is able to infer most of the associations between DCM and its causative genes by exploiting only the literature prior to their first appearance.

The Case Based Reasoning system has been tested on both projects with different objectives: in INHERITANCE with the aim of grouping a set of simulated DCM patients on the basis of their features. This test has shown the CBR system capability to cluster patients with similar diagnoses on the basis of their closeness. In ONCO-i2b2 the purpose was to enhance the project's data warehouse's query tool in order to extend the patient selection process on the basis of the semantic distance score. The actual aim of this evaluation was to show how the Case Based Reasoning system can be integrated within a data warehouse query system currently used in the clinical and research practice.

The performed evaluation showed promising results; in particular, it has become clear how the systems flexibility in properly managing the data affects the many different environments in which the developed techniques are applicable. In fact, the systems have been used to perform several tasks and proved a high degree of adaptability; moreover, the techniques adopted and their actual implementation grant also a high degree of extensibility. For instance the Literature Mining system can be integrated with new Information Extraction pipelines in order to extract new types of concepts from the scientific literature, while the very nature of the Case Based Reasoning system, with its case representation based on UMLS, allows new cases from new sources to be added to the overall reasoning environment.

Future directions of the research described in this thesis comprehend: the extension and refinement of the Text Mining techniques employed in the Literature Mining system to allow the extraction of more complex data patterns, the evaluation of new distance algorithms in order to allow the system to consider different types of features (e.g. clinical measures and genomic data). Moreover, the evaluation of the CBR system must be carried on, in particular to test its performance when comparing patients

characterized by a number of features that largely overcomes the one that it has been possible to test until now. Alongside this methodological effort, it is necessary to continue the implementation of the system Graphical User Interface in order to properly integrate all the functionalities of the two systems and to complete a tool that can be actually used in the clinical and research practice.

Contents

Chapter 1

Introduction

Translational research is a hard-to-define context and several attempts to characterize its objectives, competencies and goals have been made; in general, it is commonly accepted that translational research is aimed at integrating different types of research from different disciplines with the long-term goal of improving the public health. In order to give a clear definition of translational research it is necessary to present the research types it is aimed to integrate.

The main goal of *basic* research is to acquire knowledge without bounds to its possible applications in practical ends; in [1] it is given the following definition where basic research is opposed to applied research:

> *<<Basic research is performed without thought of practical ends. It results in general knowledge and an understanding of nature and its laws. This general knowledge provides the means of answering a large number of important practical problems, though it may not give a complete specific answer to any one of them. The function of applied research is to provide such complete answers.>>*

In contrast to basic research, in [2] it is given the following definition of clinical research as the combination of three areas:

1. *Patient-oriented research. Research conducted with human subjects (or on material of human origin such as tissues, specimens and cognitive phenomena) for which an investigator (or colleague) directly interacts with human subjects. Excluded from this definition are in vitro studies that utilize human tissues that cannot be linked to a living individual. Patient-oriented research includes: (a) mechanisms of human disease, (b) therapeutic interventions, (c) clinical trials, or (d) development of new technologies.*
2. *Epidemiologic and behavioral studies.*

3. *Outcomes research and health services research.*

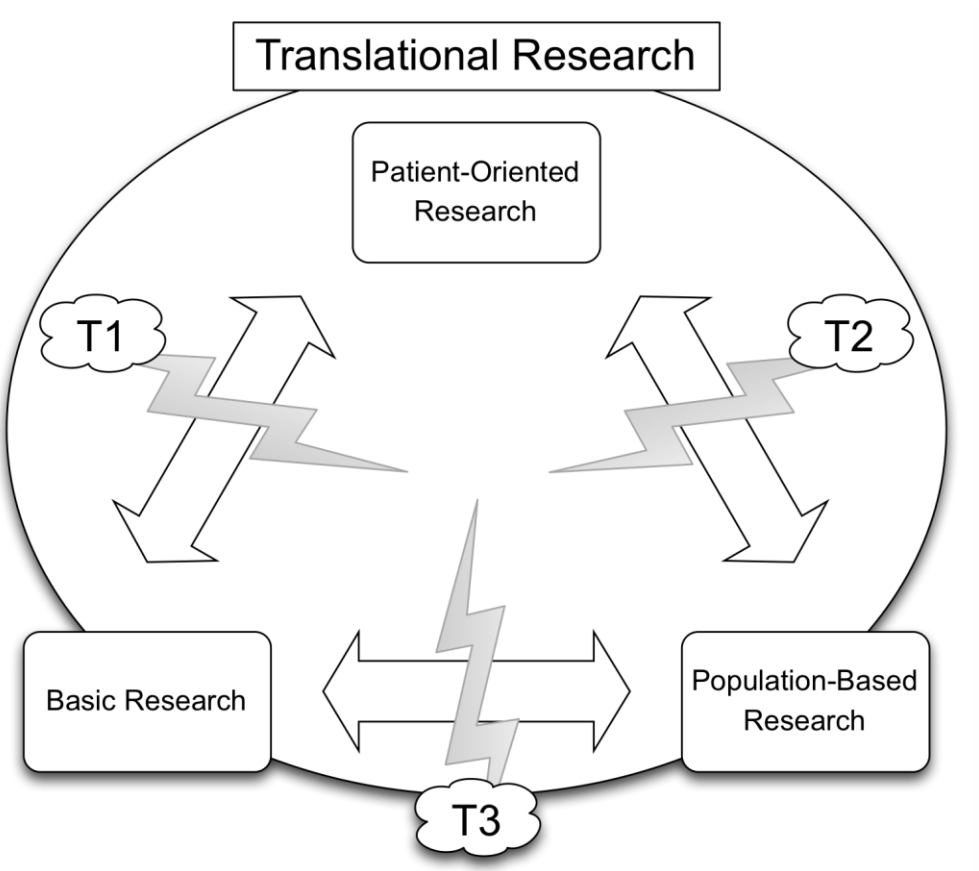

Figure 1.1: Graphical representation of the translational research workflow.

The definition of translational research is not as clear as the definitions of the types of research it is aimed to integrate. Figure 1.1 shows the three types of research areas it integrates and highlights the activities (T1, T2 and T3) that allow the translation of results between them. The working definition in [3] states that:

> *<<Translational research fosters the multidirectional integration of basic research, patient-oriented research, and population-based research, with the long-term aim of improving the health of the public. T1 research expedites the movement between basic research and patient-oriented research that leads to new or improved scientific understanding or standards of care. T2 research facilitates the movement between patient-oriented research and population-based research that leads to better patient outcomes, the implementation of best practices, and improved health status in communities. T3 research promotes interaction between laboratory-based research and*

population-based research to stimulate a robust scientific understanding of human health and disease.>>

The proposed model of translational research is graphically represented in Figure 1.1; from this representation one can infer that the research process is a continuing cycle between the three integrated research types, in order to underline that there is not a real end in the process, but the results of each stage can, and rather have to, affect the others. Another important trait of the translational research process (represented in Figure 1 by the bi-directional arrows) is its multi-directionality that overcomes some previous definitions (e.g. [4]) where the three stages followed each other in a precise order (T1-T2-T3) and allows the maximum flexibility.

Another important aspect inferable from Figure 1.1 is that, while the patient-oriented research area is included in the translational scenario, basic and population-based research are only partially included; this means that patient-oriented research issues are completely translatable into clinical practice and public health, while the other two address some issues that are not translational.

1.1. Knowledge Management in the Translational Scenario

The research activity described in this thesis has been conducted within this emergent scenario; the main goal of the present work concerns the investigation of knowledge management techniques and the development of software solutions to support the information exchange between the several actors operating in the translational research context. In particular, the focus of this research activity has been set on the management, analysis and exploitation of the data coming both from basic research, such as the impressive amount of scientific literature available, and from patient-oriented research, such as the even larger volume of patients data gathered by healthcare providers and researchers.

The research efforts have been divided in two main directions giving rise to the development of as many systems: a Literature Mining system for the extraction and proper management of the biomedical entities contained in the published literature and a Case Based Reasoning (CBR) infrastructure that assesses the degree of closeness of medical cases (i.e. patients) on the basis of the multiplicity of their features.

Both these aspects are affected by a noticeable data heterogeneity: scientific literature contains many types of entities relevant for biomedical research (e.g. genes, proteins, drugs, diseases) while every healthcare organization typically manages its own patients' data autonomously according to the codification systems that better suite its specific purposes. In order to overcome this issue and, moreover, to exploit all the potentialities of the available data, the leading thread of this work has been

flexibility. The Literature Mining system has been designed to be flexible enough to extract and properly manage many relevant biomedical entities and, moreover, its architecture allows to easily extend this analysis to any other concept type. On the other side, the Case Based Reasoning system proves its flexibility by allowing to collect patient cases from disparate sources and by avoiding to force them to conform their feature set to a fixed model.

The Literature Mining system deals with the enormous amount of data coming from the published scientific literature. It accomplishes several tasks: from the retrieval of the raw data (i.e. the articles) from PubMed to the actual Information Extraction process of the relevant entities from their abstracts and the successive storage of the achieved information in a persistence layer flexible enough to manage the many different entities extracted. Moreover, a Literature Based Discovery system has been developed in order to exploit the data extracted from the literature and point out hidden relations that could lead to the discovery of new knowledge.

The Literature Mining system can be used for different purposes: the summarization of a given set of articles through the list of relevant entities contained in their abstracts, the retrieval of a scientific articles set according to the concepts cited into their abstracts and the discovery of potential new knowledge through the identification of indirect associations reported in literature.

The Case Based Reasoning system performs several tasks to compare medical cases dealing with their heterogeneities: it maps the incoming patient data onto a common representation framework based on the Unified Medical Language System (UMLS) and exploits a distance score algorithm to represent these cases in a space where their relative degree of closeness can be evaluated. The developed algorithm produces a semantic distance score between two cases represented by a variable set of features. Another task accomplished by the Case Based Reasoning system is the generation of a query that can be submitted to the Literature Mining system, or directly to PubMed, on the basis of the feature set of a patient, thus allowing a patient-oriented literature interrogation.

The Case Based Reasoning system can be used both for identifying, given a new case, the subset of known cases that prove to be more similar to it according to the distance algorithm; and to achieve the distance matrix relative to a given set of cases in order to identify clusters on the basis of the descriptive features of each case.

Finally, a Graphical User Interface (GUI) has been developed to provide the access to the functionalities of the two systems. For the moment only a limited set of functionalities is made available through the GUI:

- browsing and interrogating PubMed with *ad hoc* queries generated with the support of medical terminologies (UMLS in particular) and

- exploiting the distance score algorithm on the available known cases and linking the patient-domain to the literature-domain with patient-oriented PubMed queries.

1.2. Outline of the Thesis

As pointed out in the previous section, this thesis is focused on knowledge management techniques applied to the translational biomedical research scenario; in particular, the primary concern of this work has been the investigation of Information Extraction approaches, to exploit the medical knowledge available in the scientific literature, and Case Based Reasoning techniques, to manage efficiently real patients data.

The dissertation is organized as follows:

Chapter 2 briefly introduces the biomedical informatics techniques suitable to cross the translational barriers presented in the previous section; moreover, particular focus is given to translational bioinformatics techniques.

Chapter 3 examines the Text Mining methods and the biomedical terminologies available with a particular focus on those exploited in the Literature Mining system; moreover, other Information Extraction systems are analyzed and, finally, the Literature Based Discovery technique is introduced.

Chapter 4 introduces the Case Based Reasoning (CBR) paradigm, its typical workflows and the related issues; afterwards, different types of similarity metrics are analyzed and, finally, CBR systems applied to the biomedical context are presented.

Chapter 5 describes the overall architecture of the system developed in this thesis and, after a detailed discussion on the main technologies adopted, focuses on the developed methods and the actual implementation of the two main systems purpose of this work: the Literature Mining system and the Case Based Reasoning system. Finally, the system Graphical User Interface is presented.

Chapter 6 discusses the results of the testing phase of the systems in the context of two translational projects.

Finally, **Chapter 7** presents my concluding remarks and examines some future research directions.

Chapter 2

Biomedical Informatics in the Translational Scenario

This chapter introduces the biomedical informatics techniques particularly suitable to cross the translational barriers presented in Chapter 1. These approaches can support and speed up translational research along its entire workflow in order to achieve a better integration in the translational continuum, from bench to bedside to public health.

The three barriers to the translational integration of different types of research are respectively obstacles to: the results exchange process between basic research in biology and patient-oriented research (T1), the translation of results between bedside research and population based research (T2) and, finally, the direct cooperation between population research and basic research (T3).

2.1. Technologies to Cross the Translational Barriers

The use of biomedical informatics techniques is considered to be a powerful solution in order to enable researchers in different fields to cross these barriers; in particular both emerging approaches (purposely developed for these tasks) and existing methods (originally developed with different purposes) have to be considered.

The approaches shown below are very important and largely studied in order to exploit their peculiarities in the translational research continuum; the present dissertation is, obviously, not exhaustive, but focuses only on a set of methods that have been evaluated in this work.

Much of the efforts in biomedical informatics to enable translational research belong to two distinct categories: bioinformatics techniques

(Section 2.2) focus on crossing the T1 barrier while clinical research informatics techniques are aimed at crossing T2 and T3 barriers. As stated in [5] both bioinformatics and clinical informatics scopes have a double nature, they meet two distinct needs: the user-centric need to serve stakeholders like biologists and clinicians and the knowledge-centric need to serve researchers at the bench, bedside and community; bioinformatics approaches should, for example, help in the identification of cellular regions in order to target them in the clinical practice (user-centric approach) and to help a better understanding of the molecular basis of the disease (knowledge-centric approach); clinical informatics could be used to improve patient care (user-centric) by making available and by integrating relevant information coming from different areas (knowledge-centric). Both the approaches are important to achieve results in the connected scenario of translational research.

Decision support techniques are aimed at orienting the decision making process of a given stakeholder by managing the information about the environment where the decision has to be made and by applying an intelligent filtering process on the different applicable solutions [6]. Decision support systems (DSS) execute the following core activities:

- Knowledge acquisition, which involves the gathering from heterogeneous data sources of relevant information for the decision making process, actually all the data from the decision environment that could affect the decision taken.
- Knowledge representation, that is aimed at mapping the acquired information onto some structure (e.g. controlled vocabularies and *ad hoc* implemented structures) in order to make them comparable and exploitable by the system.
- Inference, which deals with the analysis of the structured data characterizing the problem using, for example, rule-based mechanisms or probabilistic models, with the aim of producing a set of possible decisions to be made.
- Explanation, that describes the proposed set of decisions and the process that led to their choice.

DSS have found appliance in all the fields of biomedical informatics; in bioinformatics several DSS have been proposed to support decisions taken by bench biologists; for example in [7] it is presented a classification-based DSS that operates on gene regulatory sequences. Some systems described in literature focus more on translational aspects: [8] and [9] describe two systems that can provide a set of possible clinical decisions by considering genetic information along with phenotype data. DSS have also been implemented and evaluated in the clinical informatics domain [10].

The paradigm of translational research imposes to DSS designers to update their standard paradigms in order to make their system able to help crossing the translational barriers; a system with these features spans

different disciplines and force its stakeholders (that obviously come from different areas) collaborate and communicate efficiently; to this end the scientific community is looking with interest to the incorporation of "Web 2.0" technologies inside the DSS to increase and improve the communication between experts in the different areas involved in the decision making process [11].

The effective design of DSS suitable to be integrated inside the translational research continuum, independently from the technologies chosen for the actual implementation, could lead to essential tools for supporting researchers coming from different disciplines in a efficient decision making process.

Natural language processing (NLP) systems belong to two different subfamilies: natural-language understanding systems that extract structured information from non-structured (or, at least, semi-structured) natural language sources (e.g. texts) in order to make them usable in other applications; natural language generation systems, on the other side, perform the inverse task and generate human-understandable language from structured data coming from a wide range of applications. NLP is tightly related with computational linguistic techniques that model the phenomena related to natural language.

In bioinformatics many efforts have been made to design NLP systems that could lead to biological knowledge extraction from texts; in clinical informatics the focus has been set on both the extraction of information from unstructured or semi-structured clinical notes and on the automatic generation of clinical documents, such as discharge letters, that translate a set of structured data into a human-understandable text.

The need of NLP systems is becoming deeper and deeper; in fact, scientific literature is rapidly growing [12] and the increasing adoption of EMRs will also lead to an increased amount of natural language sources to be analyzed. In particular, for the translational issues, NLP techniques can link structured and unstructured data coming from different phases of the translational research continuum.

The management of data belonging to different sources is a crucial aspect of translational systems that, for their own nature, use information coming from different scientific areas; these data are often represented in a multitude of different formats and, therefore, methods to standardize information in order to make them exchangeable and usable together face a particularly difficult task; standards are thus a fundamental topic in the translational research scenario.

Standards in biomedical informatics are first of all ontologies and vocabularies [13], but also best practices to use and incorporate them in biological, clinical and public health contexts. Some examples of well-known biomedical standards are Health Level 7 (HL7) [14], defined to guarantee the interoperability of health care applications, Systematized Nomenclature of Medicine-Clinical Terms (SNOMED CT) [15], a clinical

terminology, suitable to represent information associated with patient care and the International Classification of Diseases and Related Health Problems (ICD) [16] that is widely used in the public health context, but not only. Many efforts have also been spent in crossing the barrier of disparities between data coding in order to allow a consistent re-use of already produced data [17].

The Unified Medical Language System (UMLS) [18] is a meta-thesaurus managed by the US National Library of Medicine (NLM); in UMLS, that contains a wide range of vocabularies from different disciplines, all the concepts (that may also belong to more than one vocabulary at the same time) are mapped on a single Semantic Network, thus establishing relations between a wide spectrum of areas (from molecular level to population). In the following UMLS will be treated with more details.

Information retrieval techniques are aimed at retrieving from one or more databases those data that satisfy some criteria expressed as a set of queries; the quality of this task can be measured in terms of precision and recall. Many efforts have been spent in the biomedical informatics context in order to integrate different data sources to be queried as a homogeneous system. Relevant resources in this field are, for example, PubMed, the online retrieval tool for accessing the MEDLINE citation database [19], and, on the bioinformatics side, a collection of different information retrieval systems that work on biological data [20] and try to integrate them in order to consider more than one resource simultaneously.

In the clinical informatics field the attention has been set on the development of information retrieval systems to extract the data of patients corresponding to a particular clinical context or the most relevant literature related to the context of interest [21].

The integration of information retrieval techniques with NLP algorithms and knowledge representation can lead to the development of more user-friendly information retrieval systems based on the "question-answer" paradigm [22].

Information retrieval is inherently a translational technology because attempts to identify relevant information across multiple heterogeneous sources and, in this new translational context, researchers are working on both adapting old information retrieval paradigms to the new application perspectives and evaluating the use of emerging technologies like "Semantic Web" [23].

Electronic Medical Records (EMRs) are the computerized evolution, due to the spread of data in electronic format in the health care, of paper medical charts; EMRs contain, for each patient, a set of data that can be recorded by the primary care providers (e.g. physicians and nurses), but can also come from services such as pharmacy and laboratories. As already said, data coming from EMRs are fundamental for the development of effective DSS and their integration is very desirable in terms of

translational research. The research in the EMRs field could also lead to the development of better clinical and research data warehouses [24].

In bioinformatics, in order to achieve an affective genotype-phenotype correlation, the problem of integrating genomic information inside EMRs is very studied and many solutions have been proposed (e.g. [25] and [26]).

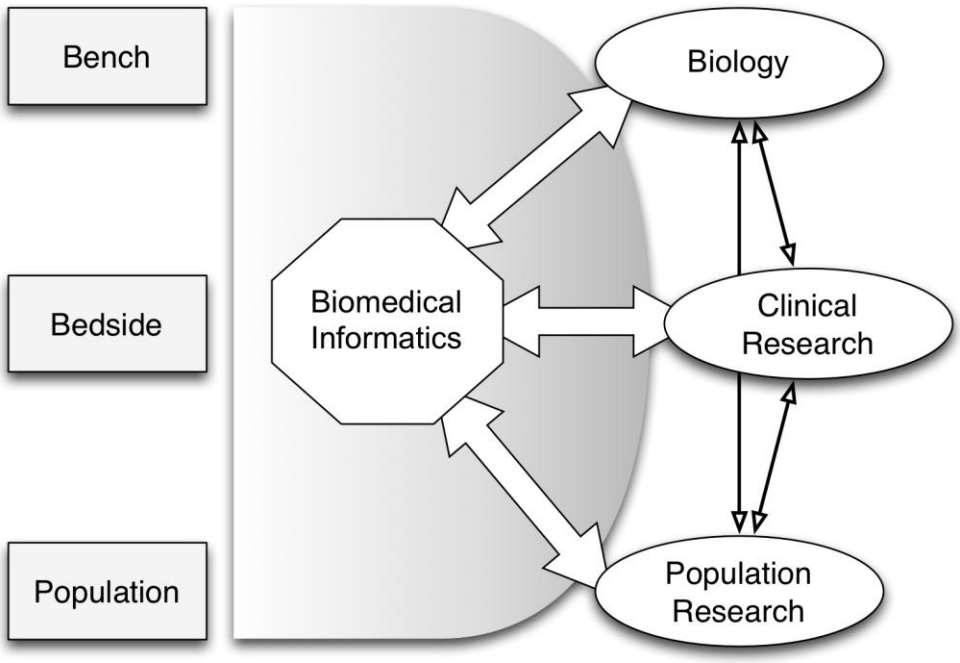

Figure 2.1: The central role of biomedical informatics to support translational research throughout its phases.

The very nature of translational research makes the presented technologies fundamental to cross the barriers to the integration of many different disciplines from bench to bedside to population; in Figure 2.1 it is shown the central role of biomedical informatics in supporting the different phases of translational research.

2.2. Translational Bioinformatics

In this section the application areas of translational bioinformatics identified in [27] will be introduced; each area will be briefly described and its peculiarities, recent advances and opportunities will be pointed out.

In the health-related genomics field many opportunities for translational bioinformatics have recently appeared since the sequencing power has undergone a continuous increase while costs drop constantly. The first task to be accomplished is the creation of the information systems able to store

and process this larger and larger amount of data that tends to stress the available information infrastructures. Another important field is the one concerning the creation of algorithms for associating measured genetic variations with clinical domain outcomes such as disease risk and therapeutic response; for common genetic variations, where a good statistical power is ensured, this is not a particularly though task, but in the case of rare variants this field needs the development of new methods.

Another interesting task in health-related genomics is the genome-informed medicine that asks the full genome sequencing for all patients to be exploited in health-care systems in order to prioritize disease surveillance and choose the best treatments in terms of cost and efficiency.

In the field of health genomics recent advances have been made, for example, towards: a) the integration of genetic risks into the clinical routine workflow [28]; b) the use of EMR-based phenotyping to create controlled cohorts for clinical trials, instead of more expensive specially created study cohorts [29] and c) the use of self-reports with phenotype information (hence not collected by investigators) for genetic association studies [30]. Other areas of appliance are, for example, the exploitation of molecular networks to interpret genetic data, the integration of genetic information in large-scale epidemiologic studies and gene patenting issues.

In the field of molecular diagnostics and prognostics the main goal is to use RNA expression, protein expression, small molecules and metabolites data to identify biomarkers associated with clinical syndromes. This task covers the basic profile-based clustering of cellular populations characterized by molecular data, the use of these data to identify subclasses of diseases and the use of molecular data to provide information on disease risk and therapeutic response, because of their better correlation with phenotypes than simple morphology information.

Some recent works in the field of molecular diagnostics and prognostics are, for example, GWAS studies aimed at identifying potential biomarkers [31], rare genetic disease investigations [32], identification of model systems that prove to be good analogs of human phenotypes for specific diseases [33].

The global collective knowledge of biomedicine is contained in published literature; PubMed, the main online resource for (bio)medical literature, contains more then 21 millions of references to articles with (almost for each one) the abstract accessible free of charge. The main goal for bioinformatics is to structure this large amount of natural language data, for example mapping the contained concepts onto a selected terminology, in order to make them automatic searchable and indexable. One example of open task in this field is the efficient identification of gene names within texts; in fact genes often have names that involve some common words and this makes the automatic identification task very probing. From the translational point of view NLP techniques can create associations between concepts extracted from literature (both clinical and molecular concepts) or

identify connections between different disciplines by analyzing their separate literatures.

Another source of unstructured texts to be analyzed are health-care-worker notes from EMRs; the analysis of these data is even more difficult than the analysis of scientific literature, in fact these texts are not subject of editing and often contain abbreviation and errors, but, on the other side, the results of their automated analysis, when integrated with other patient-related information (e.g. lab values, imaging, structured information from EMRs) could lead to great advantages in the clinical practice, for example to better understand the clinical responses to diseases and therapies. These advantages could even increase if the biobanks data were integrated in order to connect the clinical and the molecular domain.

Some recent experiences in the computational analysis of text are the creation of drugs side effects from the analysis of FDA labels [34], the attempt of creating a system to analyze large text repositories, such as PubMed, in order to extract concepts coming from controlled vocabularies and link them together [35] and the development of a classifier to predict drug-drug interaction and the underlying mechanisms, starting from PubMed data.

Translational bioinformatics can also be applied in the context of systems medicine. Systems medicine, like system biology that focuses on entire biological systems and not just on their single parts, drives attention to human biology in order to characterize diseases with molecular data as well as on therapeutic interventions. Similarly, system pharmacology adapt this approach to drug response analysis, focusing not on a model where a drug is considered only for its main target, but where the target is the entire human system; systems pharmacology is a naturally suitable area for translational bioinformatics because drugs are both clinical and molecular entities. System medicine and system pharmacology need translational bioinformatics to develop and validate models that prove to be useful and predictive.

Some experiences in these fields are the creation of a complete human metabolic network from literature and genome [36], a complete network of human diseases and genes [37] and a computational embryonic stem cell regulatory network from multiple databases, publications and software packages [38].

Public health records and population-based information resources are a data source suitable for connecting basic biology to medicine; for example they can be useful to detect and to help understanding of drug-related mechanisms (like drug effects and interactions) or to perform population surveillance for communicable diseases (like influenza). In this field new sets of data are emerging from health social networks where patients report signs, symptoms and, in general, medical issues autonomously so that it is possible to monitor changes in population health with statistical techniques.

Another effort in public health applications is the informatics for integrating biology and bedside (i2b2) system, that can connect several clinical data warehouses to support clinical cohorts finding and linking molecular and clinical data [39]. Other works in this area regard the need of making internet data, and in particular health related data, understandable for computers; in [23] it is proposed a solution that exploits semantic web technologies in order to map knowledge among entities of interest (like genes, diseases, drugs, etc.); in [40] the authors consider the opportunities and challenges for cloud computing in translational bioinformatics.

Since translational bioinformatics uses also individual patient information that can be linked to genomic data and phenotypic measurements, very important issues are those related to privacy and ethics; some recent works on these topics are [41] and [42].

On the opposite side, it is eventually needed to make the results of translational studies reportable back to the participants; this is known as the de-identification issue [43].

Chapter 3

Text Mining

Text Mining (TM) can be defined, in contrast with Data Mining that deals with already-structured data, as the process of extracting structured knowledge from unstructured sources and, in particular, from textual documents [44]. Thus the main goal of TM techniques is to make the knowledge contained in texts explicit and to provide it, in a concise form, to the users. TM can be divided in three main activities:

- *information retrieval* collects relevant resources (i.e. texts) from the available sources
- *information extraction* performs the main task of the whole TM process in order to achieve, from the unstructured sources, the contained information in a structured form
- *data mining*, considered as the final step of the TM process, is aimed at finding associations between the data gathered with the information extraction techniques.

In other words, TM allows extracting precise information from unstructured data sources in order to produce potential new knowledge; the latter task requires also, once new hypotheses generated from text have been formulated, an experimental validation process by the experts.

TM techniques applied to the biomedical domain have to face with the strong multidisciplinary nature of this filed; in fact, biomedicine is characterized by many interdisciplinary aspects and by scientific communities with different needs and views on the same knowledge space. This is a founding aspect of the whole biomedical scenario and, therefore, it is not possible to disregard it but, on the contrary, it has to be managed properly. Moreover, also TM itself is highly connected to different disciplines: from Natural Language Processing (NLP) to Information Extraction (IE) and data mining.

A field very connected to biomedical TM is that of biomedical terminologies and ontologies; in fact, the information extraction task in this field cannot be carried out properly without the vast amount of knowledge contained in the disparate terminological resources; these data are used to recognize concept occurrences in the text and to characterize them with additional information actually gathered from those resources. This mapping procedure performed on the biomedical terminologies is challenged also by the high degree of term variations in biomedicine, such as abbreviations; this leads the biomedical scientists to develop solutions different from the ones adopted for standard language processing. Moreover, in order to be effectively usable, these resources must grant a certain degree of internal consistency, a wide coverage of the domain, interoperability and, finally, they should be developed, if not for the specific purpose of being exploited by TM systems, at least to be compliant with them. For this reason TM techniques can also be used to support the update of biomedical terminologies and ontologies and to extend their coverage.

TM techniques, thanks to their flexibility, can provide the analyzed texts with added information; in fact, such techniques are not limited to the extraction of concepts and relations from their sources, but can operate also in the opposite direction, that is the characterization of these textual resources with additional information gathered, for example, from biomedical ontologies, in the form of metadata.

This chapter is structured as follows: in Section 3.1 the basic NLP techniques are presented, in Section 3.2 a brief survey of the resources available to be exploited by TM techniques is performed, in Section 3.3 the problem of IE in biomedicine is analyzed and in Section 3.4 the Literature Based Discovery approach is introduced.

3.1. Basic Natural Language Processing Techniques

The analysis of texts in natural language, with the aim of examining its structures and components at different levels of detail, is a complex task usually performed in several sequential procedures instead of in a unique overall step. It is commonly accepted that NLP analysis can work on a text at three different levels [45]:

- *lexical level*, where the single words are considered;
- *syntactic level*, where groups of words (i.e. sentences or clauses) are taken in exam;
- *semantic level*, where the meaning beneath the text in analyzed.

For the purposes of the present thesis, only some tools belonging to the first two levels will be treated; for a more complete dissertation about NLP, refer to [46].

Considering the overall NLP process as the collection of its atomic components is not sufficient to achieve a complete picture; in fact, to reach the goal of a complete content-oriented text analysis, an integration of the different components is needed.

3.1.1. Text Tokenization

Text tokenization is typically the first step of any NLP system, it works on the lexical level of the analyzed text and is basically aimed at identifying its basic atomic units, called tokens. The basic strategy in order to achieve the tokenization of a text is to consider tokens as those characters sequences separated by blank spaces or punctuation symbols. Although this strategy could work in most cases and is often executed as a "first attempt" by many tokenizers, it presents a series of problems that have to be solved:

- periods do not always mark the end of a sentence; this is the case of abbreviations, usually separated by periods. This issue is made further complex when the abbreviation happens at the end of a sentence and, therefore, the last period accomplishes the double task of delimiting both the abbreviation and the whole sentence.
- apostrophes, usually denote in English a possessive marker (i.e. Saxon genitive) and should therefore be treated adequately.
- hyphenated words (e.g. *time-consuming* or *co-occurrence*) put the problem of how they should be addressed, whether to separate them in two tokens or combine them in a single comprehensive token.
- many entities are represented in texts with different formats and tokenizers must face this heterogeneity. Examples are: numbers, dates and addresses.
- in most cases sentence-delimiting marks are periods, question marks and exclamation marks, but, sometimes, other symbols are used for this purpose (e.g. colon and semicolon).

All these issues typical of general English texts are even more complex when the NLP system has to deal with biological texts where non-standard punctuation and orthographic patterns are very common [47]. Text tokenizers are typically based on regular expression matching [48], on lexicons or on hybrid approaches that exploit both strategies.

3.1.2. Morphological Analysis

Morphological analysis is aimed at grouping many possible variants to their canonical form, called *lemma*; this kind of processing is useful, in

particular, for specific types of words, such as verbs and nouns that typically undergo inflection. This type of process can effectively uniform the content of a document at a lexical level.

Two families of approaches have been implemented in order to achieve this goal: approaches that do not exploit a lexicon, called lexicon-free approaches, and morphological analyzers that use these resources. The most widely used morphological analyzers are based on the lexicon-free algorithm described in [49]. This approach is very simple (it is based on the sequential appliance of specifically developed regular expressions) and nevertheless has turned out to be a standard in this field; anyway, for non-standard tasks, approaches that exploit special-purpose lexicons are required.

The other family of approaches to morphological analysis is based on a lexicon. Two types of lexicons can be considered:

- *full-forms lexicons* that explicitly enumerate all the possible variants of the same canonical form and that allow the analyzers based on them to perform a simple lookup process,
- lexicons composed exclusively by canonical forms which possible suffixes require the analyzers to perform a more complex task to match the words contained in the text.

3.1.3. Part-of-Speech Tagging

Part-Of-Speech (POS) tagging assigns to each token the relative grammatical class; this operation is performed in order to disambiguate the tokens at a syntactic level. This task cannot be completed at the lexical level of the analysis (i.e. considering the single words in isolation); in fact many words can be associated with different POS tags depending on their role in the phrase (e.g. the word "*still*" can be an adjective, an adverb, a conjunction, a noun and even a verb on the basis of the specific context where it is used). Therefore, to assign each token to the proper grammatical class it is necessary to consider the local syntactic context in which the token is included, that is the tokens that come before and after it in the sentence.

POS taggers can be divided into two families, both representatives of supervised approaches, as they need an annotated corpus to be trained:

- *Rule-based taggers* apply a set of pre-determined rules to the context which the tokens belong to; the context is constituted by the words themselves (lexical level) or, since the POS tagging process is iterative, by the temporary POS tags assigned to them in previous iterations of the POS tagging algorithm. A widely adopted rule-based tagging algorithm is described in [50].
- *Statistical taggers* typically base the tagging process on the calculated probabilities for the token to be associated to a specific

grammatical class given the *n* previous POS tags assigned (*n-gram* approach [51]). Other approaches adopted for statistical POS-tagging are based, for instance, on maximum entropy [52] and support vector machines [53].

Since most available POS taggers are based on supervised and data-driven approaches, it is necessary to deal with the problem of which annotated corpora to use; in fact, POS taggers are typically trained on general purpose English annotated corpora, so their performance on biological texts, when trained in such a way, noticeably worsens compared to the one on general English documents. In [54] it is shown how a special purpose biomedical corpus can boost the POS tagging performance for medical texts.

3.1.4. Text Chunking

The chunking process is aimed at identifying within a text special phrases; an example of such phrases is *Noun Phrases*, subsets of consecutive words that represent a nominal entity and are constituted by a nominal head and a series of pre- end post-modifier referred to this head. Chunking algorithms, as well as POS tagging ones, adopt the supervised approach and therefore need an annotated corpus in order to be trained.

Several algorithms have been proposed to face the chunking problem: the most widely used is the rule-based algorithm described in [55], but also approaches based on hidden Markov models [56] and support vector machines [57] have been attempted.

Identifying noun phrases within text has proven to be a useful task in pre-processing documents from which concepts, such as Named Entities, have to be extracted; in fact, most terms are contained within such phrasal units [58].

3.1.5. Typical NLP Architecture

A generic description of NLP cannot be limited to the dissertation of the particular text processing techniques, but it has to deal also with the way these separate modules are put together in order to create a working system. Typically, a system aimed at extracting knowledge from unstructured texts performs a cleansing process of the input sources in order to uniform the text that will be analyzed; in fact, the source documents can belong to many different formats, such as PDF, HTML or plain text.

The first text processing technique to be applied is the tokenization of the input text; this operation is generally followed by the morphological analysis aimed at identifying the lemmas of each token. Afterwards, the POS tagger assigns all the tokens to their grammatical class. Optionally, POS taggers may also exploit some advanced analysis such as the recognition of specific named entities and of acronyms in order to achieve better performance in the tagging process. The typical output of a POS

tagger is constituted by a set of tuples where each token is associated with the right POS.

At the end of these single-word analyses, the NLP systems proceeds with the processing of phrasal entities and, in particular, the extraction of noun phrases; this operation typically exploits the POS tag assigned to the tokens. Biological named entity recognition can be based on such an analysis, which ends with the noun phrases recognized within the source text.

The operation of mining named entities from text deeply relies on the associated biological resources, such as lexicons, ontologies and thesauri.

3.2. Biomedical Resources

Text Mining (TM) systems in biomedicine are aimed at extracting from text those entities that specifically belong to the biological and medical context; to perform such a task these systems rely on biomedical resources, such as terminologies and ontologies [59]. Two are the main tasks faced by TM in this field: entity recognition, the identification of biomedical concepts within text, and relation assessment, the evaluation of the relationships that exist between the mined entities [60]. Although the most frequently analyzed types of sources are generic molecular biology texts, also biomedical literature [61] and clinical narratives [62] have received many attentions by the TM community.

The entity recognition task relies on terminology resources such as:

- the International Classification of Diseases (ICD) [16],
- the Medical Subject Headings (MeSH) [63],
- the Systematized Nomenclature of Medicine – Clinical Terms (SNOMED-CT) [15],
- the UMLS Specialist Lexicon [64].

In general, biomedical terminologies are created and maintained to collect all the names associated to a given entity employed in the biomedical domain.

Biomedical ontologies, on the contrary, are aimed at studying entities of biological significance and the relations that occur among them. This distinction, however, is valid just in theory; in fact, the distinctions that are applicable to real-world biomedical resources is not so clear. Ontologies, although focused on the main task of tracing relations among concepts, often contain a set of synonyms and, on the other side, terminologies often are not limited to the collection of different names referred to a concept, but their hierarchical organization brings also some information on basic relations between the concepts (typically "*is a*" relations).

In the entity recognition process, while a simple lexicon (i.e. a collection of words used in the domain) can lead to the identification of the biological terms, terminologies and even more ontologies, allow to detect if two

recognized words refer to the same biomedical entity. The capability of extracting from free-text the relevant entities, whichever form is used to represent them, and to map them on an environment enriched by a set of relationships that can occur between the mapped entities, makes the UMLS Metathesaurus and the UMLS Semantic Network (that will be treated extensively in Section 5.1.1) a good solution for biomedical entities extraction.

The relation assessment process follows the entity name recognition step and aims at identifying the relations that actually link the entities in the source text. The TM system is therefore asked to recognize, beyond the entities, also those parts of the text representing a potential relationship between two or more concepts. These relationships can be identified on the basis of lexical items, syntactic structures and patterns and can involve more than one sentence. Such complex analysis requires advanced computational linguistics approaches which dissertation goes beyond the scopes of this thesis, where TM techniques have been used to develop an entity recognition system.

In the following a non-exhaustive list of biomedical resources for TM (based on those described in [60]) is given.

3.2.1. Lexical Resources

Two widely used lexical resources are:

- *WordNet* [65], a lexical resource developed at Princeton University; it is structured as a collection of synonyms referring to different concepts described by a definition. It is organized into linguistic categories (e.g. nouns, verbs, adjectives). WordNet (2.0) contains more than 111.000 entries and it is integrated with some ontology-like features. Its coverage of the biomedical field is limited and, therefore, it has not been used in many projects in this area [66].
- *UMLS Specialist Lexicon*, that provides lexical information for NLP systems applied to biomedicine. Each record is characterized by syntactic (e.g. POS), morphological (e.g. lemma) and orthographic (spelling variants) features. It can be described as a general English lexicon enriched by many biomedicine-specific terms. Since the Specialist Lexicon is only a part of the whole UMLS system, it does not contain information about the synonyms and the semantic relations between its records; in fact, such information are reported by the other two main resources of the system: the UMLS Metathesaurus and the UMLS Semantic Network.

3.2.2. Terminological Resources

Another type of resources for biomedical TM are terminological resources, which gather the different names by which a biomedical entity is referred; they are typically, but not necessarily, tied to a specific domain, such as genes, protein or drugs. Some terminologies are organized in a hierarchical-like manner and most of them belong to the hierarchical class of trees, where records are linked between each other by parent-child relations (i.e. "*is a*" relations). Terminologies play an important role in biomedical entity recognition. Three examples of such resources are:

- *Gene Ontology* (GO) [67], annotates gene products for model organisms. It is divided into three separate hierarchies organized by: molecular function, biological process and cellular component. These hierarchies are not trees, but acyclic graphs where the nodes represent the terms and the edges are "*is a*" relations. In general, GO terms are characterized by a set of synonyms and by a textual definition.
- the *Medical Subject Headings* (MeSH) [63] are a controlled vocabulary of medical terms widely used to index and to search biomedical documents. It counts about 23.000 terms (called *descriptors*) organized in 15 hierarchies. Each descriptor, excluded the 15 hierarchies roots, is linked with at least one parent descriptor. MeSH are characterized by a broad scope, in fact they span the vast majority of sub-fields of biomedicine, and by a poor granularity.
- the *UMLS Metathesaurus* is one of the three resources developed by the National Library of Medicine (NLM) in the U.S. within the UMLS system. It is a collection of biomedical vocabularies linked to each other thanks to a common codification system onto which entries from different sources are mapped; an extensive dissertation of the Metathesaurus is performed in Section 5.1.1.1.

3.2.3. Ontological Resources

While terminologies are mainly focused on the collection of different names referable to a specific biomedical entity, biomedical ontological resources are not mainly concerned with names, but with classes to which the entities belong and the relations that can occur between these classes. In practice, since terminologies often link the entities with "is a" relationships and since many ontologies collect also the synonyms referred to the concepts, the distinction between these two classes of resources is not always so clear. In the following two ontological resources are presented:

- the *Systematized Nomenclature of Medicine – Clinical Terms* (SNOMED-CT) is a biomedical terminology developed according to the description logic formalism [68]; it is formed by 18 hierarchies and its concepts are characterized by several descriptors, such as alternative names, and different relations (that go beyond the basic "is a") with other concepts.

- the *UMLS Semantic Network*, the last of the three UMLS resources presented in this chapter, consists in a set of Semantic Types, namely subject categories which the concepts in the Metathesaurus belong to, and a set of Semantic Relations that represent the possible relationships that can connect two Semantic Types. The Semantic Network is extensively described in Section 5.1.1.2.

The UMLS system comprehends all the three types of biomedical resources for TM (lexical, terminological and ontological) and, given its wide coverage of almost all sub-fields of this area, has proven to be a very powerful resource. For this reason it has been chosen, for the purposes of the work described in this thesis, as the basic resource which the Literature Mining system (Section 5.2) and the whole CBR System (Section 5.3) are founded on.

3.3. Information Extraction in Biomedicine

Information Extraction systems in biomedicine perform the recognition of biomedical entities' names and their further extraction from texts in natural language; such entities can be, for instance, genes, proteins, drugs or diseases. The task of recognizing concepts names and associate all their names variants is often referred as (biomedical) Named Entity Recognition (NER) [48]. Systems performing this task go beyond the simple term recognition because, after this necessary step, they link the recognized term to the relative concept inside the biomedical resource used in support of the extraction process. Despite many resources are available for almost all the subfields of biomedicine (Section 3.2), these resources do not address all the issues connected with the recognition task.

In this section, the extraction of genes and proteins from biomedical texts will be considered; in fact, the same techniques can be adapted to work with other concepts extraction tasks and, in particular, the recognition of these two entity categories presents some issues that will be discussed. Gene and protein names share many characteristics that make their recognition a non-trivial task:

- they often contain special characters such as uppercase characters, digits, hyphens and brackets;
- they can suggest some characteristics of the gene/protein such as function, species and similarity to other molecules [69], but IE systems cannot relate exclusively on them to derive such characteristics [70];
- they can appear in many different forms, such as abbreviated, plural and anaphoric [71].

IE systems cannot disregard these aspects in order to achieve effective results.

Some of the issues connected with the recognition task of such entities are [48]:

- *ambiguity* is probably the main issue relative to the extraction of gene and protein names; in fact, it often happens that the same name is used to denote different genes and proteins (i.e. there is not a 1:1 ratio between names and concepts). Another case of ambiguity is due to the many names that are also common English words (i.e. *can*, *for*, *white*). Moreover it can happen that a name changes the referred molecule over time [70].
- In [72] it is shown that a simple name-matching strategy to recognize genes within a text leads to very poor performance and, therefore, approaches that exploit lists of these ambiguous names or use statistical models to solve this issue have to be adopted.
- *synonyms* denote the different names ascribable to the same entity; it often happens that a name can be referred to the various homologous genes or proteins across different species. A list of synonyms for each retrievable entity is therefore needed and, moreover, IE systems should manage an update strategy of such lists with new names that regularly appear referred to already known genes and proteins.
- *variations*, especially character level variations, lead up to a failure rate of 79% in the gene name extraction task [73]. These types of variations are due to:
    - insertion or deletion of special characters
    - exchange of indices
    - replaced words
    - omitted words
    - different words order
    - abbreviated words
- *newly discovered genes and proteins* add a further complexity to the already complex task of IE; in fact, once a new molecule is discovered, the registration process and the relative update of the available resources is far from immediate. It is therefore necessary to develop recognition techniques for names of new genes and proteins discovered.
- *the purposes of the IE system* also affect the actual need of facing the listed issues; for instance, an IE system aimed at extracting protein-protein interactions must recognize also proteins cited in indefinite phrases, such as "*this protein…*", because the interaction may be explicated in the document between that phrase and the name of another protein. On the other side, a system aimed only in the extraction of proteins cited within the text can overlook the indefinite

phrase because the name of the protein is certainly explicated before in the text.

Several approaches can be attempted to perform the IE task; the methods used by these approaches can be grouped in four families:

- *dictionary-based methods* that perform a simple matching process with a biomedical resource;
- *rule-based methods* that exploit manually- or automatically-constructed rules to match concepts within text;
- *machine learning methods* that use statistical models, developed with machine learning techniques, to perform the recognition task;
- *hybrid methods* that exploit two or more of the above methods while performing the concept recognition.

3.3.1. Dictionary Based Methods

Dictionary-based methods depend on the available biomedical resources, relative to the specific type of entity that the system is aimed to recognize. The simple matching between parts of the text and the selected resource(s) has been shown to lead to poor results [74] mainly because of the issues discussed above.

In [75] the authors adapted BLAST [76], a biological sequence comparison algorithm, to deal with morphological variations in gene and protein names. The proposed approach translates the characters of the names, both of the molecules in the reference resource and of the mined text, into nucleotide sequences and tries the matching process on this transformed space.

In [77] a new dictionary of gene and protein names is created merging HUGO [78], OMIM [79] and UniProt [80]; this "meta-resource" is curated with semiautomatic methods in order to increase the natural tolerance of its source vocabularies to the issues of IE for molecule names.

In [73] a two-phase method is described: first a simple matching process with a protein database (extended with probabilistic techniques) is attempted, then a Naïve Bayes classifier is used to filter the result set.

3.3.2. Rule Based Methods

Rule-based methods extend the types of variations manageable by the IE system based on dictionary matching; in fact, while a dictionary-based approach can deal exclusively with character- and word-level variations, a rule-based approach manages also word-order and syntactic variations.

The method described in [81] uses manually developed rules that rearrange text in the neighborhood of some core terms in order to achieve a set of strings that potentially represent a gene/protein name. These core

terms are words that often occur in molecule names. Once the set of strings is done, the matching process with the biological database is performed.

In [82] the authors show a manually-curated context-free grammar [83] composed by more than 160 rules that are applied to the component terms extracted from known protein and enzyme names in order to encompass as many variations as possible.

3.3.3. Machine Learning Methods

Machine learning methods overcome rule-based methods when names belonging to still-not-considered conventions have to be recognized. For instance in [84] it is described an IE system that exploits Hidden Markov Models (HMM) trained with a manually annotated corpus to classify sequence of words and determine if they represent a protein name.

In [85] the authors show a method to create training corpora for gene recognition systems. In order to create such resources they exploit a curated list of genes and the set of articles where they are cited. In [71] another method based on HMM is shown; in particular, in this work the authors have developed a model with so many features (morphological, syntactic and semantic) that they faced the problem of the lacking of a training corpus wide enough to train adequately the whole model; the proposed solution exploits *k* Nearest Neighbors techniques to solve this problem.

3.3.4. Hybrid Methods

Since all the discussed methods present specific advantages, but also some lacks, hybrid methods try to make a synthesis of several approaches in order to achieve overall methods with better performance. In [85] the authors present a method that combines statistical and knowledge-based approaches by applying both automatically- and manually-generated rules in order to extract gene names from text.

In [86], finally, it is shown an approach that combines all the strategies described so far (dictionary-based, rule-based and machine learning) to recognize protein names within a source text.

3.4. Literature Based Discovery

Literature Based Discovery (LBD) is a technique suitable in translational research to support the researcher in the process of connecting the very sparse and huge knowledge available in the scientific publications in order to extract potential new knowledge.

However, nowadays the available scientific knowledge is very huge, increases very rapidly and usually is specialized on very restricted domains. This could make difficult considering the complete literature

relative to a specific domain or, more ambitiously, linking efficiently the existing knowledge related to different domains and finding new relations. LBD supports the researcher in the discovery of unknown relations among scientific knowledge exploiting text mining techniques applied to the scientific literature. The goal of such a process is the generation of new hypotheses representing potential new scientific discovery.

LBD technological approach was first introduced by Swanson in [87]; the main goal of that research was searching automatically in a large set of documents the connections that can be inferred between relevant concepts, but not explicitly reported in the literature. The knowledge discovery process results from the combination of existing knowledge and observations in a novel way capable of obtaining evidence of new hypotheses. The LBD process is based on two distinct entities: the concepts relevant to the research domain and the available literature, that is the set of documents related to the domain that potentially refer to these concepts. Usually the documents are public scientific papers, and the concepts are medical terms mentioned in the documents, but not directly connected to each other.

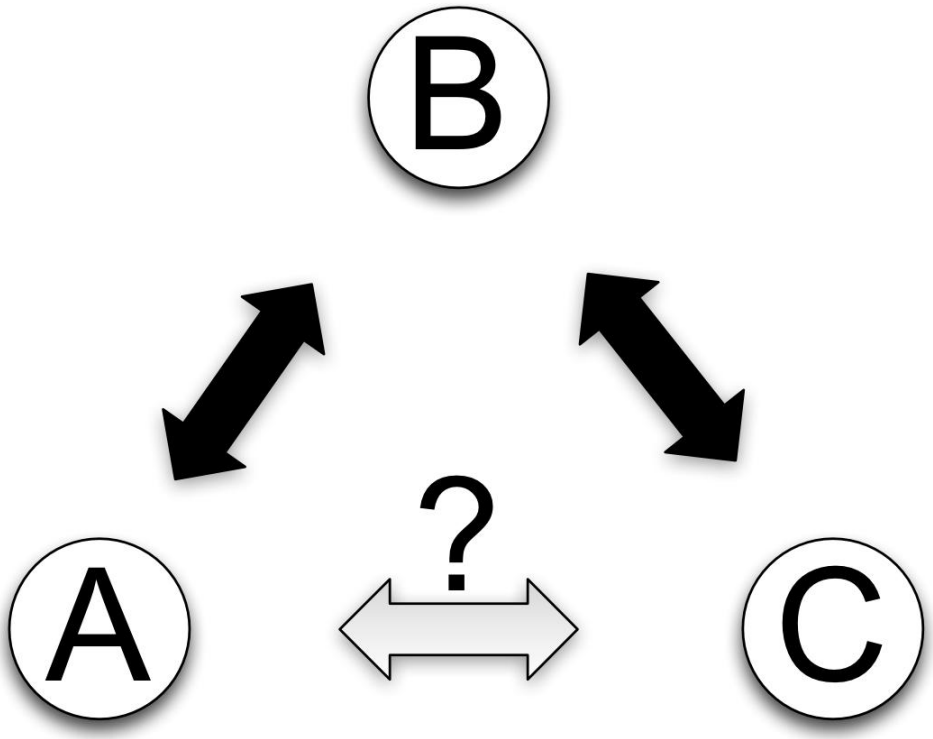

Figure 3.1: The ABC model for new knowledge discovery introduced by Swanson

Definition: Two literatures L_A and L_C referring respectively to the set of concepts *A* and *C* are disjointed if there is no overlapping between the two sets *A* and *C*.

The theory for new knowledge discovery introduced by Swanson, called ABC model, is defined as follows (Figure 3.1): if A and B are related, and B and C are related, it follows that A and C might be indirectly related through B. The discovery process can be carried out in two different ways, called Open Discovery and Closed Discovery; the first one is used to discover new connections, while the second is used to confirm potential new hypotheses.

In particular, the open discovery process performs three sequential actions:

1. given a literature L_A related to a concept A (e.g. a disease), extracts from L_A the set of concepts B directly related to A
2. obtains the literature L_B relative to some B concepts; this set of concepts is obtained after a filtering process in order to restrict the analysis to specific entities (e.g. disease effects)
3. extracts from L_B the interesting concepts C, excluding the ones already known as related to A. Concepts C (e.g. substances used to treat such effects) are related to A through B, therefore these relations can potentially represent new knowledge.

The Closed Discovery process searches in the literatures L_A and L_C for the concepts B related both to the concepts A and C. The validation of a new hypothesis consists in finding among the concepts B, those justifying the relation between A and C.

3.4.1. LBD Systems

After the Swanson work, many systems for automating the LBD process have been proposed. They can be distinguished on the basis of the discovery paradigm adopted. The first one is Arrowsmith [88], a tool for LBD based on the Closed Discovery model. It identifies the common words or phrases in the titles of two disjointed literatures. These terms are then ranked on an estimated relevance probability useful to filter out the not interesting concepts. Although it has been the first effort to automate the discovery process, many steps require a lot of manual intervention. A recent version of the tool exploits MeSH terms to rank B-terms also on the basis of the domain context of the articles.

The system proposed by Gordon & Lindsay [89] analyses complete MEDLINE records to compute several lexical statistics, such as word frequency counts and record counts. Such statistics are used to rank the documents and assess their relevance in the discovery of hidden

connections. The system allows to manually filter out less relevant concepts and manage synonyms and generalizations.

Weber et al. have developed a system called DAD (Disease-Adverse Reaction Drug-Drug) [90], based on the Open Discovery paradigm, able to extract interesting concepts mapped onto UMLS, instead of simple words, from literature. The filtering of B concepts can be based on the semantic types defined in UMLS.

One of the most interesting systems is the one developed by Pratt and Yetisgen, called LitLinker [91], that couples the Open Discovery model to a data mining algorithm to automatically extract an ordered list of intermediate concepts B to be filtered out on the basis of different criteria (too much general, too much cited, not belonging to a specified semantic type). Finally, the system clusters the similar concepts and, using the Apriori algorithm [92], finds association rules between the starting concept and the clusters (B concepts). After proper filtering of the rules based on their support, the same process is applied to achieve, from B concepts, the final concepts C.

A more recent algorithm proposed by Srinivasan [93] combines different lexical statistics to weight the terms, maps them on UMLS and, finally, introduces MeSH based profiles to properly rank the concepts.

One of the system that is more similar to the one developed in this thesis (Section 5.2.6) is Bitola [94], developed by Hristovski. In Bitola each record is represented by the MeSH terms indexing the article and the gene names and symbols found in the title and in the abstract. The association between two concepts A and B is scored on the basis of its support (number of articles containing both A and B) and confidence (percentage of articles containing both A and B over the total number of articles containing A).

As it can be guessed by considering the efforts in the field of LBD, the most critical aspects of the discovery process are:

- the choice of the knowledge sources (article titles, abstracts, MeSH terms, etc.) that, in some cases, can be affected by human errors (e.g. in manually indexing articles) or cannot be publicly available or can generate high volumes of spurious associations;
- the concepts representation problem (in many contexts it could require specific procedures for acronyms disambiguation, synonyms management);
- the design of a robust validation process (for new knowledge discoveries, a lot of time could be needed to confirm their validity).

The system described in this thesis (Section 5.2.6) is based on the open discovery paradigm that allows good flexibility in setting up the different steps of knowledge discovery processing, depending on the specific domain. It is also able to manage concepts generalization/specialization through medical concepts mapping based on UMLS. The concepts ranking is based on the support and confidence of the relations found in the literature.

Chapter 4

Case Based Reasoning

Case Based Reasoning (CBR) is a problem-solving process based on past experience. Differently from an approach that tries to solve a problem by applying a set of general rules, in CBR the main knowledge source is a set of stored cases that describe prior episodes; thus, new solutions are adapted from solutions of old cases that have proven to be similar to the one in exam. This approach is corroborated by the role of reminding in the human reasoning process [95]. CBR is different from many AI approaches because, instead of using general domain knowledge (e.g. the case of expert systems), it is able to exploit the knowledge coming from concrete previously experienced problems.

Two axioms support the adaptation of a CBR approach to solve real-world problems [96]: *"similar problems have similar solutions"*, thus solutions to already solved problems are useful to build up a solution to an incoming "not-yet-faced" problem; and *"problem types tend to recur"*, therefore future problems are expected to be similar to old problems. In a world where these two axioms are true, CBR is an effective problem-solving strategy. In particular the second axiom doesn't forbid considering similar, although not identical, cases as a useful resource to solve an incoming problem.

Another important aspect of CBR approach, besides the *reasoning by remembering* paradigm, is the fact that incoming cases, solved thanks to the use of information coming from stored cases, become, in turn, part of the knowledge base that will be used in the future: *reasoning is remembered* [96].

4.1. Introduction to CBR

The development of CBR techniques is driven by two different motivations: the need to model human behavior and to improve the performance of artificial intelligence (AI) systems. The way that CBR adopts to follow these goals is to enhance AI systems' performance inspired by the human problem-solving abilities, that evolve over time and work with hard problems in a limited knowledge environment. Therefore CBR can be applied to deal with these main problems:

- Knowledge acquisition: rule-based knowledge systems present the main problem of defining the rules on which the system works; in fact, these rules may be difficult to be identified and, in order to model the reasoning process of an expert, it is difficult to state if a set of rules is complete or needs to be improved. Furthermore these rules can amount to very large, hard-to-manage numbers.
- CBR systems adopt a completely different approach and found their reasoning process on past experience, extracted from past cases; thus it is not necessary to analyze and decompose the problem-solving process in order to infer the specific rules of the context. Obviously some domains need more efforts to be the context of a CBR approach; in fact, cases may be unavailable or difficult to be accessed, like cases described in natural language. This kind of cases need a case engineering effort in order to be used in a CBR workflow; information must be identified, properly represented and extracted from available data.
- Knowledge maintenance: the process of designing a solution for a given problem is typically recursive and the solution identified in the early phases generally needs to be refined; furthermore also task requirements may be updated over time, making the previous solution finding process obsolete. CBR systems grant big advantages in this knowledge maintenance task because they, by their very nature, allow new cases to be added to the system without expert intervention. Furthermore, CBR systems can be deployed with a limited set of evaluated cases that will be eventually expanded only if the initial configuration of the system proves to be insufficient to solve the problem.
- Finally, a fundamental aspect of knowledge maintenance in CBR systems is their need to deal only with cases' types that actually occur in practice, while rules-based system need to take into account all the possible problems that can occur in order to generate a coherent set of rules.
- Problem-solving efficiency: building up the solution for a new case on past experiences over already solved cases, instead of repeating the solving effort, increases the efficiency of the reasoning system.

- Quality of solutions: rules may be imperfect, especially for domains whose dynamics are not completely understood; in such areas, solutions derived with a CBR strategy may be more accurate that solutions coming from the execution of a chain of imperfect rules, in fact cases reflect what really happens in the real world.
- User acceptance: problem-solving systems are successful only if users accept their conclusions; especially in health related task, reliability of the reasoning process is a critical aspect and systems whose strategy is not directly explainable (e.g. neural network systems) add a further obstacle to their pain acceptance by the users' community. CBR systems are based on prior knowledge often generated by their stakeholders and the whole reasoning process can be explained with human-understandable artifacts (e.g. the data on the solved cases used by the system).

According to [97] CBR can be described as a cyclic process involving "the 4 R's" Retrieve, Reuse, Revise, Retain, that is:

- RETRIEVE the case, or the set of cases, that prove to be similar to the case in exam.
- REUSE the information contained in these retrieved cases in order to solve the problem relative to the case in exam.
- REVISE the solution produced.
- RETAIN the useful parts of the case in exam and the proposed solution in order to improve the case base and to achieve better performance in future problem-solving tasks.

4.1.1. Interpretative and Problem-Solving CBR Systems

CBR systems can be divided in two main categories: interpretative CBR systems and problem-solving CBR systems that respectively address the problems of classifying new situations and to suggest solutions to new problems using the knowledge collected over time.

An interpretative CBR systems works in order classify a new case exploiting the information available about cases that have already been classified. This kind of techniques has found appliance also in diagnostic systems where the current symptoms of a patient can be considered as his characterizing features and can be used to compare the case with data about patients that have already been diagnosed (i.e. classified).

Interpretative CBR systems' workflow typically involves four main steps: situation assessment, case retrieval, case comparison and case re-using. The first step is needed in order to select, among the whole set of features available for the cases to be considered as prior knowledge, which are the ones that really matter in order to build up an effective CBR system. The second step involves the retrieval, among the entire base of prior cases, of those that are compatible with the features that have been selected after

the situation assessment step. At the third step the reasoner compares the case in exam with the cases selected by the case retrieval task and, on the basis their features and classification, proposes a classification for case in exam. The final step is aimed at storing the case in exam, its features and the proposed solution (obviously if accepted by the user) in the case base, in order to make it available as prior knowledge for future interpretative processes.

Problem-solving CBR systems have the goal to apply prior solutions in order to generate the solution to an incoming, not-yet-faced, problem. In such a CBR system the first three reasoning process steps executed are the same as in interpretative systems: situation assessment, case retrieval and case comparison. In addition, problem-solving CBR systems evaluate how similarities (and differences) between the case in exam and prior cases can be taken into account in order to determine a new solution that fits prior knowledge. This step requires the proposed solution to be adapted from the previous solutions.

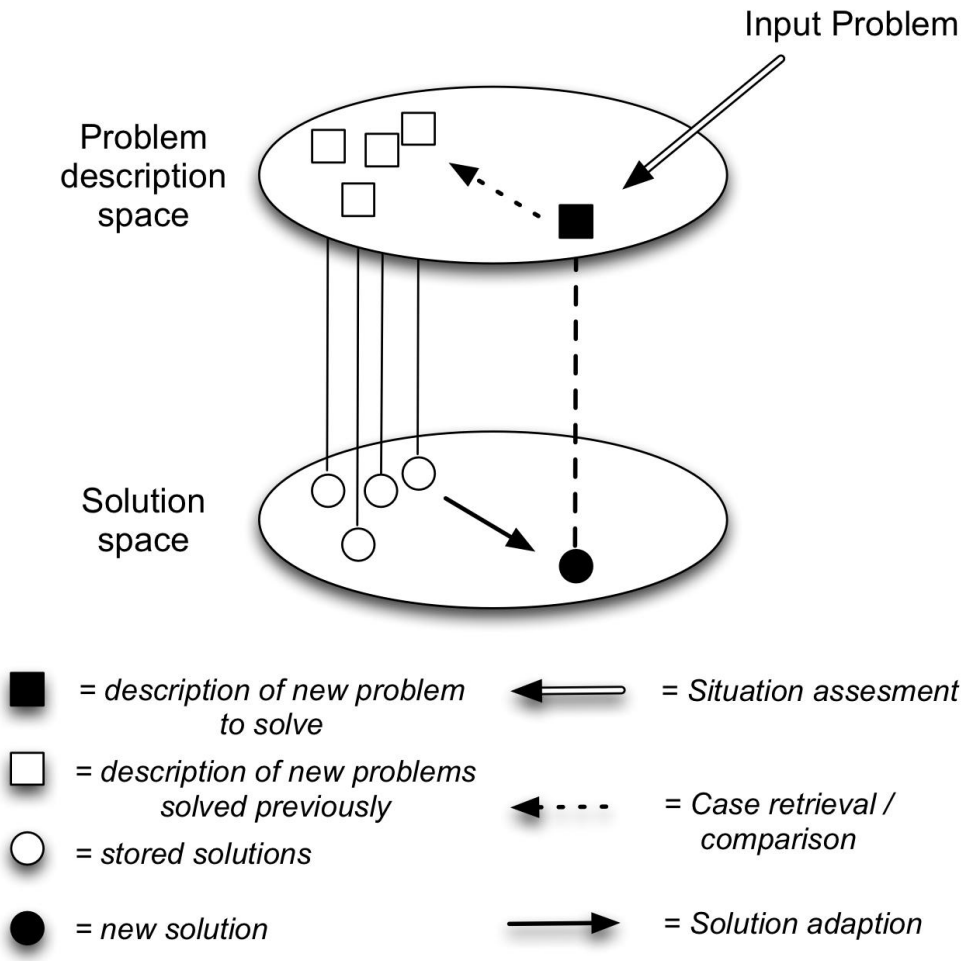

Figure 4.1: The overall process of a standard problem-solving CBR system.

While for interpretative CBR systems there is only a similarity metric to evaluate, i.e. the similarity between the features set of the compared cases, problem-solving CBR systems exploit the relationship between two different kinds of similarities: similarity evaluated in the problem description space (i.e. features) and the similarity that works in the problem solution space. This process is described in Figure 4.1 where, after the situation assessment step, cases are compared considering their features; the solutions of problems that prove to be similar in problem description space are then used as the starting point for generating a solution for the incoming problem. This whole strategy is clearly based on the *"similar problems have similar solutions"* axiom. Information to perform such a task is contained in the descriptions of evaluated cases but also in the similarity metrics and adaptation strategy.

4.1.2. Connected Areas and Relevant Issues

In the following section some areas tightly connected with CBR and the work presented in this thesis will be described.

Analogical reasoning techniques are, from some points of view, similar to CBR techniques; in fact both solve new problems by interpreting prior episodes that have some kind of analogy with the one in exam. Analogical reasoning and CBR share the same cognitive model. Nonetheless these two approaches differ because analogical reasoning techniques are traditionally focused on abstract knowledge and structural similarity while CBR techniques often find application in systems that have to deal with specific cases with specific solutions.

In addition analogical reasoning techniques do not take into account the processes that occur before and after the analogical mapping of the considered cases (i.e. their mapping onto a space where they can be compared), while CBR does. For example the case retrieval step, that is a fundamental part of both interpretative and problem-solving CBR systems, is a process that analogical systems avoid by considering the cases to be compared as an input. Another aspect that most of the analogical reasoning systems don't take into account is the re-use of incoming cases in order to extend the case base available. Thus analogical similarity techniques can be considered as a fundamental part of the broader CBR scenario.

Both information retrieval (IR) and CBR systems perform a retrieval step, but for the latter this process is more active; in IR systems delegate most of the problem of defining the right query to the user, while CBR systems can start from a set of features that don't have to be exactly the same that will be found in the retrieved cases; this flexibility in the retrieval step is due to the situation assessment step that CBR systems perform before the case retrieval.

Another important difference between the two approaches is the fact that IR systems work with an exact matching strategy, this means that retrieved information is completely coherent with the constraints of the performed query (built on the basis of the case in exam); CBR systems, on the other side, work with a "most similar" strategy and therefore they return the case or the set of cases that prove to be similar to the one in exam, given a specific similarity metric. Thus, retrieved cases in CBR systems may present conflicts with some of the attributes that were specified in the retrieval query generated from the case in exam. Therefore, while in IR systems the retrieved cases depend only on the case in exam, in CBR systems they depend also on their peers (i.e. the other prior cases that act like competitors in the retrieval step).

Nevertheless IR systems can be used to improve performance of the CBR retrieval step and the information they store can potentially add new cases to the prior case library.

CBR systems differ from traditional inductive learning systems, based for example on symbolic and neural network approaches, because these techniques found their working strategy on examples that are not singularly taken into account when this strategy is put in practice. In fact they use these examples only in the training phase, while CBR systems, at each run, use directly the entire set of specific prior cases.

Retaining specific cases presents many advantages. First it makes the decision more explainable and verifiable because the system can point directly to the cases supporting its decision and the user (whether a human or another system) can directly access to the prior cases set in order to assess their applicability. Second it is useful to resolve conflicts; for example, in an interpretative CBR system, it may happen that the two most similar previous cases to the one in exam lead to two contradictory solutions. In this case, in order to decide which solution to follow, it is useful to access directly and evaluate the cases in the context where the decision has to be taken. In traditional inductive learning systems the advice coming from these two conflicting cases would be combined in order to provide a single answer and the conflict itself would be hidden to the user.

Another important aspect that differentiates these two systems' families is that CBR systems perform an incremental learning, while traditional inductive learning systems need to be trained again when the prior case base changes. Furthermore, the performance of CBR systems is correct, on its case library, also if this library is composed by a reduced number of cases; this aspect allows the set up of prototype CBR systems that, if needed, can incrementally expand their case base with time.

On the other side inductive learning systems can work also in knowledge-poor environments, while CBR systems need the knowledge to be available (at least in part) in order to properly define similarity and retrieval criteria.

In the following some issues connected with CBR process will be described.

The most basic problems in CBR are connected with the first two steps of the overall reasoning workflow: situation assessment and case retrieval. In fact, the further steps will succeed only if the prior cases taken in exam are the relevant ones and, in order to retrieve the best data, it is important to perform a good indexing of the available set of cases. Among the techniques used to solve this problem we can mention: inductive learning methods, instance-based learning techniques, introspective reasoning and explanation-based techniques [98].

Once the best prior cases have been found it is possible to proceed with the case-comparison step. Nearest neighbor techniques are usually used in order to provide a case or, more likely, a set of cases, that prove to be similar to the unsolved case in exam; in general, the matching process between two cases is not perfect because the values of the features of the new and previous cases are usually different. Moreover, it may happen that some of the features belonging to the selected feature set, have not a value in one or both the compared cases, i.e. there are missing values in the description of the two cases. Therefore the usual approach to compare two cases is to define some proper similarity metric. (Similarity metrics will be discussed in Section 4.2).

An additional challenge that similarity metrics may face is to take into account the different importance that different features may have in the specific context; in some situations it is possible to develop weighted similarity metrics but, in general, CBR systems perform a "flat" comparison and try to solve this problem with the situation assessment step where irrelevant features are discarded from the considered feature set. Anyway the information about cases already in memory and the knowledge coming from experts can be exploited to develop a similarity technique that dynamically weights the features considered in the matching process.

Another possible difference between different CBR systems is the way they represent the information about cases; in fact, while most systems represent these data as a collection of feature/value pairs, it is possible to develop a more structured representation, like a graph representation. Obviously a more complex data representation involves the development of an equally more refined similarity metric.

Uncertainty and incompleteness are also issues deeply connected with the CBR paradigm. Uncertainty is involved in the semantics of the features that describe the cases, in the evaluation of the similarity scores obtained by comparing these features and in the solution adaptation phase for the problem-solving CBR systems, while incompleteness is faced when the usually sparse space of prior cases is taken into account and in the description of each case. Fuzzy logic techniques are a good solution in order to deal with this kind of imprecision.

Fuzzy logic techniques may help to characterize imprecise and uncertain information about cases in the case retrieval step [99], while fuzzy matching is useful in order to evaluate partial matching in case comparison and to adapt the solutions with fuzzy combination rules [100].

Therefore, fuzzy logic may be involved in many aspects of the CBR process:

- The definition of the case base itself may be considered as a fuzzy process because the cases it contains cannot be considered simply "completely useful" or "not at all useful" but their usefulness, depending on the problem addressed, is a matter of degree.
- Fuzzy logic helps representing cases whose attributes have imprecise values.
- Fuzzy techniques can be used in order to retrieve appropriate cases.

4.2. Similarity Metrics

Quantifying the similarity between two cases is a key aspect in CBR systems; most of them assess similarity basing on similarity metrics that exploit values of the describing features. The idea of a feature/value representation of cases stands at the basis of the whole CBR paradigm and of its problem space, where the cases, opportunely mapped, are located in a position that is coherent with their distances calculated with the similarity metric itself. Recently new similarity mechanisms have emerged that are not founded on the feature space idea; some of them has arisen directly from the CBR research while others come from other areas of data analysis. This section will present the taxonomy of similarity mechanisms proposed in [101].

The standard approach of similarity in CBR assumes representing a case as a feature vector and the similarity is assessed on the basis of this vector; thus problem spaces are traditionally represented as vector spaces where is reasonable to properly manage noisy cases, to choose which features better fit the solution finding task and to apply dimension reduction techniques on the feature vectors.

However similarity metrics based on the feature vector representation are not the only strategies available; when more complex data representations are adopted, more sophisticated similarity metrics are accordingly used; for example cases may be represented with an internal structure that goes beyond the vector representation and the similarity techniques that are expected to work on these data have to deal with such complexity.

The taxonomy proposed in [101] organizes similarity mechanisms into four main subclasses:

- Direct mechanisms that work with the standard feature vector representation of cases and that is the most used strategy in CBR systems.
- Transformation-based mechanisms that re-interpret the concept of similarity considering it as the effort needed to transform one instance into the one against whom it is measured.
- Information-theoretic measures that are directly derived from information theory.
- Emergent measures arising from an in-depth analysis of the data, that apply a significant processing power in order to characterize them.

4.2.1. Data Representation

The concept of similarity is tightly connected with the representation of the data; therefore it is useful to consider the different adoptable representation strategies before considering the similarity metrics themselves.

Case representations can be categorized as follows:

- Feature/value representations
- Structural representations
- Sequences and strings

Feature/value representation is the simplest strategy, where each case is described by a set F of features characterized by a numeric value normalized in the range [0,1]. Each case (x_i) is represented by a feature vector

$$(x_{i1}, x_{i2}, \dots, x_{in}).$$

The features describing a case may be internal, when they describe the concept in terms of its attributes, or external, if they are not relative to the own nature of the case, but are produced by an external system/actor. For instance in a diagnostic system, where the cases are patients characterized by some data, features like sex, weight and blood type may be considered as internal, while the proposed diagnosis is not peculiar of the case itself and therefore is considered as an external feature.

Recently, several approaches have been presented in order to enhance the feature set of the available cases. For instance, in [102] the authors present a method to extend the representation of each individual case (textual cases in this case) by mining the training corpus, while in [103] it is presented a strategy to enhance the case representation based on Web mining techniques. Both scenarios, despite exploiting different resources in order to enhance their case representations, result in a feature/value representation.

Practical experience in the development of CBR systems has shown that feature vector representation of data is not always possible in order to effectively represent the complexity of cases encountered in practice: it often happens that data have an internal structure that needs to be represented [104] [105] [106].

Although the structural representations proposed in literature are tightly dependent on their specific context of appliance, they can be categorized in the following three families:

- Hierarchical structures, like the one proposed in [104], extend the standard feature/value representation by allowing the single feature to be represented not by a single value, but by a new vector of values (each one associated with a feature that is hierarchically contained in the top-level one)
- Network structures extend the concept of hierarchical structures, where the only possible link is the "part-of" relationship that occurs between a feature and its top-level containing feature, by allowing the presence of many types of links characterized with a rich semantics [105].
- Flow structures add to hierarchical and network structures, which are a static representation of data, the temporal dimension. An example of such a structural representation can be found in [106].

Another possible solution in order to structure the data representing the cases is to use features that belong to a taxonomy and that, therefore, also in a simple feature/value representation, bring the power of a structured representation inside the CBR system.

When cases are recorded as free-text [107], the most suitable way to represent the data is the bag-of-words strategy that come the information retrieval literature and supports similarity assessment. A bag-of-words representation strategy is often performed by removing from the free-text (that is the actual value of the data) the most common words of the specific language (called stop words) because their removal doesn't cause the informative content of the text to change significantly, while it reduces the dimension of the data and, consequently, the effort needed to process them; afterwards, the strings can be converted in a vector space representation where the single words are the features and their number of occurrences in the text is the assigned value for each feature; on such a data representation direct similarity methods can be applied.

4.2.2. Direct Similarity Mechanisms

The direct similarity mechanisms represent the most used strategy in the CBR community because they are computationally efficient and effective in most cases; direct similarity is tightly connected with feature/value

representation of data. These mechanisms define the problem space as a vector space and have to face the issue of data dimension.

The principle that stands at the basis of direct similarity mechanisms is that incoming cases are classified evaluating the class of their nearest neighbors. This technique is commonly referred as *k*-nearest neighbor (*k*-NN), where the *k* nearest cases (in the problem space) are used in determining the class of the incoming case. So *k*-NN classification systems perform their work in two stages: they determine, among the selected prior cases, the *k* that are more similar to the one in exam and then use the classes of these *k* cases in order to classify the incoming one.

Let us assume that we have a base of prior cases (x_i) described with a feature/value representation where each value has been normalized in the range [0,1] and where each case is labeled with its class *y*. The goal is to classify an incoming unlabeled case *q* by comparing it with other cases and exploiting the classes of the *k* cases that prove to be more similar to it. The distance between *q* and each case in the case base x_i, is representable as:

$$d(q, x_i) = \sum_{f \in F} w_f \delta(q_f, x_{if})$$

where *F* is the set of considered features, w_f is the weight assigned to each feature and δ is the distance function.

The set of features *F* can be chosen in order to represent the most important available features and the δ function can incorporate the domain knowledge. A basic version of the distance function is the following:

$$\delta(q_f, x_{if}) = \begin{cases} 0, & f \text{ discrete } \& \; q_f = x_{if}, \\ 1, & f \text{ discrete } \& \; q_f \neq x_{if}, \\ |q_f - x_{if}|, & f \text{ continuous.} \end{cases}$$

While this metric performs quite well for continuous normalized numeric features, it does not the same job for discrete features; in fact, nonmatching features contribute maximally to the overall distance, while matching attributes don't contribute at all. To address this issue in [108] it is proposed the value difference metric (VDM) that uses class conditional probabilities in order to refine the contribution that discrete features give to the overall distance:

$$vdm(q, x_i) = \sum_{F} w(q_f)\delta(q_f, x_{if}),$$

where $w(q_f)$ is the weight assigned to the feature and depends on the value this feature has in the new case *q*, while the distance function δ is calculated as follows:

$$\delta(q_f, x_{if}) = \sum_Y (P(y|q_f) - P(y|x_{if}))^2,$$

where ($P(y|q_f)$ is the conditional probability of class y given the feature value q_f; in other words the proportion of instances with attribute value q_f that belong to class y. The weight $w(q_f)$ is calculated as follows:

$$w(q_f) = \sqrt{\sum_Y P(y|q_f)^2}.$$

The weight will be higher for attribute values that discriminate more between the class labels; the underlying principle in the VDM distance is that two compared attributes contribute to the overall distance in the measure of their class conditional probability.

The VDM distance has been taken as the starting point for many conditional probability distances; for instance, two distance metrics based on VDM can be found in [109]. Finally, a variety of methods to determine the class of the incoming case, given the classes and the distances of its *k*-NN, have been proposed; one example is a voting technique that makes each neighbor vote for the class to be assigned with a weight that is proportional to the inverse distance to the incoming case:

$$Vote(y) = \sum_{c=1}^{k} \frac{1}{d(q, x_c)^n} \cdot 1(y, y_c),$$

where $1(y, y_c)$ returns 1 if the class labels match and 0 otherwise and n (usually set to 1), if greater than 1, reduces the influence of more distant cases in the class decision process.

4.2.2.1. Similarity and Distance Metrics

Often the terms "similarity metric" and "distance metric" are used as synonyms of an affinity measure between two objects, but in mathematics the concept of metric is well defined and a measure, in order to be defined as a metric, must conform these for axioms:

- $d(x, y) \geq 0$ (non-negativity)
- $d(x, y) = 0 \Leftrightarrow x = y$ (identity)
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, z) < d(x, y) + d(y, z)$ (triangle inequality).

Although it is possible to build a *k*-NN classifier based on an affinity measure that is not a proper metric, some optimization techniques require

to work with proper metrics [110] because they rely, in particular, on the triangle inequality.

The first distance presented in this chapter is a special case of the Minkowski Distance, in particular it is the 1-norm Minkowski distance or Manhattan distance. The general formula of Minkowski distance is:

$$MD_p(q, x_i) = (\sum_F |q_f - x_{if}|^p)^{1/p}.$$

Another important and widely adopted Minkowski distance is the 2-norm distance or Euclidean distance. Greater values of p, despite not commonly used, have been adopted in some cases. Finally, another Minkowski distance that deserves to be cited is Chebyshev distance, where p tends to infinity:

$$MD_\infty(q, x_i) = max_F |q_f - x_{if}|.$$

Chebyshev distance represents the distance in the dimension (i.e. feature) where the two examples are most different.

4.2.2.2. Set-Theoretic Measures

An alternative approach to these geometric distances are measures that exclusively take into account the feature sets that describe two cases and not the values that single features have; these measures are called set-theoretic measures and interpret similarity in terms of feature overlapping between two cases. One example of set-theoretic measure is the Tanimoto measure, described in [111]:

$$S_{T_n}(x, y) = \frac{|X \cap Y|}{|X \cap Y| + |X \backslash Y| + |Y \backslash X|},$$

where X and Y are respectively the sets of features that represent objects x and y.

Another type of set-theoretic measure is called Jaccard index [112] that measures the proportion of features that two cases share:

$$S_J(x, y) = \frac{|X \cap Y|}{|X \cup Y|},$$

this index is widely used as a validation measure in clustering [113].

Finally, the Dice similarity coefficient [114] is similar to the Jaccard index but weights more the shared features between the two objects:

$$S_D(x, y) = \frac{2|X \cap Y|}{|X \cup Y|}.$$

4.2.2.3. Taxonomies

In case of data organized with hierarchical structure representation, where features are linked by "is-a" relationships, the very structure of the hierarchy on which the features are mapped contains information about how similar two feature values are. In [115] it is proposed a measure that exploits hierarchies:

$$\delta(q_f, x_{if}) = \frac{2N}{N(q_f) + N(x_{if}) + 2N},$$

where $N(q_f)$ and $N(x_{if})$ are the number of edges that link respectively features q_f and x_{if} with their common ancestor and N is the number of edges that separate this common ancestor with the root of the hierarchy.

Some advantages of such a measure are the relatively low computational cost of counting the edges that separate two nodes inside a hierarchy and the fact that, beyond the distance between two nodes, this measure evaluates also the absolute depth at which these nodes are located (i.e. their distance from the root of the hierarchy). On the other side this measure inherits the limitation of the hierarchy on which features are mapped and, in particular, it considers all the edges as equally important; this aspect could be overtaken if the concepts were mapped onto a network where different types or relationships can be used.

4.2.3. Transformation Based Measures

Transformation-based measures represent a similarity strategy that can be used in alternative to direct similarity mechanisms; their founding principle is to define the distance between two objects as the measure of the effort needed to transform an object into the other.

The most basic transformation-based distance is the edit distance (or Levenshtein distance) that is applicable to strings and measures the number of insertions, deletions and substitutions needed to transform one string into the other.

Starting from the concept of edit distance it is possible, by adding some domain knowledge to its simple working strategy, to define new distance measures that can work effectively in very specific domains. One example of such distance measures are the ones that assess the similarity for biological sequences; in bioinformatics two different strategies have been adopted in order to solve this problem: global alignment strategy which measures the effort to transform one single sequence into the other and local alignment strategy that defines several local similarities between regions of sequences. In [116] it is presented a global alignment algorithm

that performs well when the compared sequences are similar in length and quite similar in their content. In [117] the authors show a local alignment algorithm more suitable to compare sequences that have only small regions of similarity. The performance of both algorithms depends on how substitutions and gaps are penalized.

4.2.3.1. Networks and Graphs

When the cases are represented with a structure, also the similarity measures that work on these cases have to deal with it. In [118] the author identifies two different strategies to calculate similarity between objects that are represented with a graph structure; both strategies are based on the Levenshtein distance. While the most fundamental problem addressed by graph matching technologies is the complete isomorphism of graphs, for CBR, where exact match between two graphs of features is unlikely to occur, it is more interesting to investigate the isomorphism of the sub-graphs that compose the cases' description. In [106] a graph matching measure is presented and the authors show that this approach to similarity assessment for structured cases is effective for real-world problem solving.

4.2.4. Information-Theoretic Measures

Information-theoretic measures are directly derived from studies in information theory; one of the approaches that belong to this family of algorithms is compression-based similarity. The basic idea at the basis of these measures is that two similar documents, if concatenated and subsequently compressed, should have a dimension quite similar to the dimension of their single (i.e. not concatenated) compressed versions. More formally, as stated in [101]:

> *"[...] the difference between two documents A and B is related to the compressed size of document B when compressed using the codebook produced when compressing document A".*

The compression-based approach assessment has the advantage of working on raw data (e.g. documents) and then avoids the whole feature extraction process; it is quite obvious that these techniques heavily depend on the compression algorithm used and, in general, can be applied to cases descriptions that, for their own nature, lend themselves to be considered as raw data. In [119], for instance, it is shown that compression based similarity techniques are an affective solution with cases described by texts and that their performance can even overcome the bag-of-words strategy. In [120] the authors show a compression-based similarity metric that, exploiting a specialized compression algorithm, works on phylogenetic data.

Another similarity measure borrowed from information theory is presented in [121] and extends the concept of distance calculated on taxonomies or hierarchies; in fact one of the defects of similarity techniques like the one presented in [115] is that every edge (that, in taxonomies, represents an "is-a" relationship) is considered as equally important. In [121] the author introduces the idea of quantifying the information content relative to a concept in a taxonomy with the negative log likelihood $-log(p(c))$, where $p(c)$ is the probability of observing the concept c. In this framework the similarity between two features is defined as the information content of their "most informative" common parent.

4.3. CBR in Medicine

CBR has proven to be an effective technique to be applied in health sciences. Among the reasons of the wide use of such techniques in medicine are:

- The fact that case histories have traditionally been used to train health care professionals; therefore the paradigm of solving problems basing on information derived from prior, already solved, cases fits well to this area.
- The large presence in medical literature of detailed information about experiences on real patients.
- There are many diseases whose present understanding is still not enough developed in order to make models or universally applicable guidelines available.
- Cases help to interpret guidelines, when they are available; in fact, guidelines provide a general tool to guide clinicians and need some operational information in order to be really effective; this operational information is precisely what is contained inside stored cases which, therefore, can help clinicians by complementing the guidelines.
- In general the human body, that is a very complex biological system, is difficult to be modeled. Moreover, also when an affordable sub-model is available for instance for a particular disease process (e.g. hypertension), several diagnoses can happen at the same time to cause a specific symptom; this makes the output of such diagnostic models very complex and, in general, difficult to be made complete.
- The high data intensity of medicine strongly supports the development of techniques that take advantage of this huge amount of data in order to reason on pre-existing cases or cases derived from different data sources such as scientific literature.

The vision of CBR systems in medicine is effectively described in [122]:

"A physician — a psychiatrist — once portrayed what he envisioned as the perfect computer program to assist him in his clinical work. He described representing in a database all the patients he had been treating, with all the details of their history, environment, symptoms, diseases, treatments and evaluations. He also imagined that he could get access to patient cases of his colleagues. When encountering a new patient or a new disease episode, the system would find the most similar patients and patient episodes, would show them to him, would explain the similarities and differences and how each episode was solved successfully, and finally would provide recommendations for diagnosis and treatment for the particular episode he was dealing with.".

4.3.1. Main Trends

Guided by this vision, the main trends that today characterize CBR in health sciences are:

- An integration effort in order to set the standards for case representation formalisms [123], case structure and reasoning processes [124] with the ultimate goal of making CBR systems interoperable and to define a common query language.
- The growing availability of –omics data fosters the integration of this huge amount of information into the medical CBR systems as well as the integration of –omics data into the medical profiles of each patient. Beyond the mere integration of data, CBR systems that have to deal with such data complexity, have to integrate also bioinformatics techniques that make reasoning in this complex scenario possible [125].
- Often the internal data representation of CBR systems has been developed for the specific purposes of the reasoning system itself. Today the need of sealing the gap between this representation and the patients' representations in electronic health records (EHR) is increasingly sensed; CBR systems that could take advantage of EHR will see their applicability incredibly extended. In order to achieve this goal it is important to exploit the efforts that have been made in the last decades by the medical informatics community, such as HL7.
- Furthermore, mining cases from these sources will become more and more important in order to expand the available case base and, moreover, data consolidation efforts have to be planned.
- CBR shares some aspects with information retrieval; the growing availability of information contained in texts (e.g. medical records and scientific literature) has generated the opportunity of learning cases from these unstructured sources in order to extend the available case base [126].

- Case representations will become more and more complex, this is due to the variety of data types that can be associated to a case; for instance, data can be acquired from sensors [127], images [128] and time series [129]. In this field, efforts are needed in order to adequately synthesize these data for the CBR process. Moreover, data complexity reduction processes are needed also to deal with –omics data [125].
- With the growing complexity of data it is needed to associate the results of the CBR systems to statistical scores in order to measure their affordance.
- CBR systems in medicine have as their primary goal the exploiting of past experiences in order to solve new problems but, for the own nature of their field of appliance, they also need to take into account the changing in the best practices, guidelines and treatments. Therefore past cases should be used for their informative content, but systems have to implement a long-term follow-up strategy [130] where new findings and recent developments can be added to the whole reasoning process.
- Since medicine is evolving towards evidence-based practice, the role of CBR systems, that can take into account also those information that are hardly addable to expert systems, is becoming always more complementary to guidelines.

4.3.2. The Adaptation Problem

In order to make the use of CBR systems effective in the health sciences domain some problems have to be solved: a representation form for the cases has to be designed and a suitable retrieval algorithm has to be chosen; furthermore, the system should be able to deal with an increasing number of cases, which informative content has to be added to the case base, avoiding an infinite growth of the case base itself. This can be done by clustering cases into prototypes, by removing redundant cases that do not add new informative content or by regularly maintain the case base with the help of domain experts.

However, the most important problem that CBR systems have to face is the case adaptation problem; in fact this issue strongly depends on the specific domain and on the characteristics of the application. CBR systems that are enough general and unspecialized to deal with the entire spectrum of medical cases (e.g. [131]), for their own nature, avoid this problem. In non-medical CBR applications the adaptation problem is solved by a set of specific adaptation rules created by domain experts that consider all the possible differences between new case and the cases that belong to the case base; considering the complexity of the medical domain, it is quite straightforward that this set of rules is mostly impossible to be generated for CBR systems that deal with health sciences; however, some adaptation solutions have been developed for these systems.

Some systems avoid the adaptation problem by focusing only on the retrieval task; these systems retrieve cases that prove to be similar to the one in exam and present their content to the user. Sometimes they also highlight the main differences between the retrieved cases and the one to be solved. The main reason for choosing such a solution is due to the fact that in some application domains it is too much complicated or even impossible to acquire the right adaptation knowledge; in these domains physicians typically require the CBR systems to provide the information about past cases that are similar to the one in exam, but prefer to reason on it themselves. One example of such systems is described in [132].

Another strategy to deal with the adaptation problem is the Multi Modal Reasoning paradigm that combines CBR retrieval with other techniques that provide support to the decision making process. In particular, in some applications (e.g. [133]) CBR is combined with rule-based reasoning, since rules are a common representation of medical domain knowledge and, despite being difficult to generate for too complex systems, they are effectively used for well-confined not-too-complex areas.

Rule-based reasoning and CBR can be integrated in several ways: they can be applied with a mutually exclusion strategy, where rules deal with standard or typical aspects of the problems, while CBR faces exceptions; therefore multi modal reasoning usually applies the rule-based system first and, in case of failure, exploits the CBR retrieval system to provide the user with the most similar case to the one in exam extracted from the case base delegating the user to make the final decision. Another combination of these two reasoning paradigms can be obtained by using cases to provide a suitable context to rules and, on the other side, exploiting rules to extract more general concepts in order to characterize the cases [134]. Finally, rule based reasoning can support CBR in the solution adaptations phase with some general adaptation rules [135].

A third strategy to deal with the problem of adapting both cases and their solutions, is to generalize them in order to build some prototypes; this solution tries to solve the extreme specificity of cases that, sometimes, can strongly impede the adaptation process. Moreover, this family of solutions helps to maintain a coherent structure for the case base, to decrease the storage amount by deleting the redundant cases, to improve speed performance and, sometimes, to learn new general knowledge. One example of such systems is described in [136].

4.3.3. Other Issues

In [137] the authors identify, beyond the adaptation issue, other two problems that CBR systems applied to health sciences have to face:

- Unreliability. Reliability cannot be guaranteed even with a progressive and constant growth in the number of cases that are taken into account by the system. In fact the addition of new cases will not

necessarily make the system converge towards a greater reliability and new cases add only local improvements in the system's behavior.

- Concentration on reference. CBR systems are concentrated on reference and not on diagnostic factors that underlie these references. Therefore, without at least a case in the case base that suites the description of the case in exam, the system is not able to produce a result.

4.3.4. Classification of CBR Systems in Medicine

The authors of [137] classify the CBR system in medicine into four main families:

- Diagnostic systems. Currently most of the implemented CBR systems belong to the diagnostic systems category; these applications join the physicists providing them assistance in order to improve the diagnostic process that is entrusted to them. The final development of such systems points to autonomous diagnosis systems.
- Classification systems. These systems attempt to identify the groups, which the evaluated cases belong to; this operation do not attempt necessarily to a diagnosis. Typical examples of such systems are medical image classification systems.
- Tutoring systems. Tutoring systems provide students with the access to real patient cases; such systems are strongly based on the concept of learning by examples.
- Planning systems. These systems help in solving problems that are characterized by a number of steps, such as therapy support problems.

Alongside of this further classification of CBR systems, that is valid for those that deal with health sciences, the authors of [137] have identified also some properties that help in characterizing the CBR systems developed so far:

- Hybridity. A hybrid medical CBR system exploits several AI technologies, besides CBR, in order to solve their problem. Many systems use CBR to organize data and some data-intense techniques (e.g. neural networks) to classify them. Other systems, as previously stated, exploit rule-based systems and CBR in order to take advantage from both approaches.
- Adaptation. This property describes how the CBR system deals with the adaptation problem and if uses any adaptation methods.
- Case library size. This describes both the size of the library of stored cases, but also the degree of case generalization inside the case base in order to achieve, from a certain number of cases, a prototype that summarizes their informative content.

- Autonomicity. In diagnostic systems the degree of autonomicity denotes the level of interaction that physicians are due to have with the system before and after the diagnosis is produced. A completely autonomous system would lead the whole reasoning process without any expert intervention and its results would be accepted and put in practice; actually it is rare to have such systems. The autonomicity degree implies, beyond the evaluation of the interventions needed in the result production phase, also those interventions that expert users have to make to evaluate the results.
- Constraints. Constraints are represented by reliability, that describes if the system is always operational when needed, and by safety-criticality that denotes the need for the system to provide always the right answers, because a wrong decision could potentially create dangerous situations.

In the following some systems collected in [137] will be briefly described.

4.3.4.1. Diagnostic Systems

FM-Ultranet [138] [139] is a CBR system that detects malformations and abnormalities of fetus through ultrasonographical examinations in order to detect abnormal organs and extremities. The case representation is hierarchical and object-oriented; the hierarchy counts 39 concepts characterized by attributes that span from anatomical features to medical history and general domain knowledge. Similarity between attributes is managed on the basis of their nature: some are mathematically compared and others rely, to be compared, on look-up tables defined by experts. The final product of the reasoning process is a report that contains system's findings.

The system described in [140] uses CBR techniques to optimize image segmentation according to changing acquisition condition and image quality; the system has been used to detect Alzheimer disease from computed tomography images. The case descriptions comprise images and descriptive features about image acquisition and the patient. Similarity is calculated separately for image and non-image features and finally an overall similarity score is produced.

CARE-PARTNER [130] [141] is a decision-support system applied to long-term follow-up of stem cell transplanted patients; the system is aimed at supporting the home care providers that follow up the transplant patient. The system is hybrid because it combines rule based reasoning with CBR, it adopts a three-layered fault tolerance policy. A relevant characteristic of CARE-PARTNER is its use of large knowledge base of prototypal cases that is integrated with medical guidelines.

Systems described in [142], [143] and [144] make a specific use of prototypes (i.e. generalizations occurring by grouping cases into a more general type); the authors state that using prototypal cases may help in

learning intrinsic knowledge when domain theory is weak. In practice: *"Storing new cases may improve the ability to find solutions for similar cases, but to understand the knowledge included within, generalization is needed"*.

Moreover, the use of prototypes may help in the retrieval phase, include physicians view inside a case-based system and improve the performance of the system by decreasing the number of cases. As a drawback, cases prototyping produces a loss of information in the generalization process.

4.3.4.2. Classification Systems

The system described in [130] is applied to efficiency assessment in hemodialysis. Each dialysis session is represented as a case in the system which features are collected dynamically and statically: static features are generic characteristics of the patient (e.g. age), while dynamic features are measured from time-series.

The system presented in [145] is aimed at classifying mammalian DNA sequences; cases are represented nucleotide sequences already classified in exons and introns. The similarity mechanism exploits an edit distance that takes into account insertions, deletions and substitutions on individual nucleotides.

In [146] it is shown a multi-modal reasoning system used in therapy support for diabetic patients. Authors try to overcome the fashion of using different methodologies in the same system only as extensions to one another, by integrating the different technologies in a much closer way. The implemented system solves the well known problem of singular methods, for instance the qualification problem of rule-based systems and the library dimension problem of CBR systems. Furthermore, also model-based reasoning is exploited in the system's workflow.

In [147] it is presented an image-based CBR system for airborne fungi identification; since the cases considered by the system have an extremely high biological variability, it is not possible to generalize cases in prototypes. Each case is described by attributes derived from an image processing system. The CBR system retains old and new cases in order to use the classifications for the future.

4.3.4.3. Tutoring Systems

WHAT [148] is a tutoring medical CBR system for the training of sports medicine students; this system is aimed at giving better matching exercise prescription than the rule-based approach traditionally taught by books. The cases are described on the basis of their medical history and physiological tests. The system has been applied to cardiac and pulmonary disease patients.

In [149] the authors of CARE-PARTNER propose an evolution of their system in order to help medical students to improve their knowledge by solving practice cases. The case library is composed by prototypal cases with varying levels of complexity derived from clinical pathways. The system is also capable of evaluating the students' performance.

4.3.4.4. Planning Systems

The *Auguste* project [150] aims at providing decision support for planning the ongoing care of Alzheimer's Disease patients. In its first prototypal version the system supports the prescription of neuroleptic drugs. The system is implemented as a hybrid system that exploits CBR in order to decide if the patient is eligible to be prescribed with such drugs and a rule-based system that chooses the right drug to prescribe. About 100 features manually extracted from medical charts characterize the cases. The similarity strategy to discriminate between classes of cases (i.e. "needs drugs" and "doesn't need drugs") exploits the *k*-NN matching.

In [151] it is described a system based on the *ReCall CBR shell* (ISoft, www.isoft.fr) that decides which technological devices disabled and elderly people need in their home for independent living. Cases' features are manually transcribed from handwritten reports. The system contains 10 clustered problem space groups and 14 solution groups organized in a tree in order to easily explain the decision automatically taken by the system.

Chapter 5

System Architecture: Methods and Implementation

This chapter describes in detail the methods investigated and the systems developed during the course of my PhD. The main goals of my research have been the study of advanced techniques for knowledge management and the development of new software solutions suitable to support translational research.

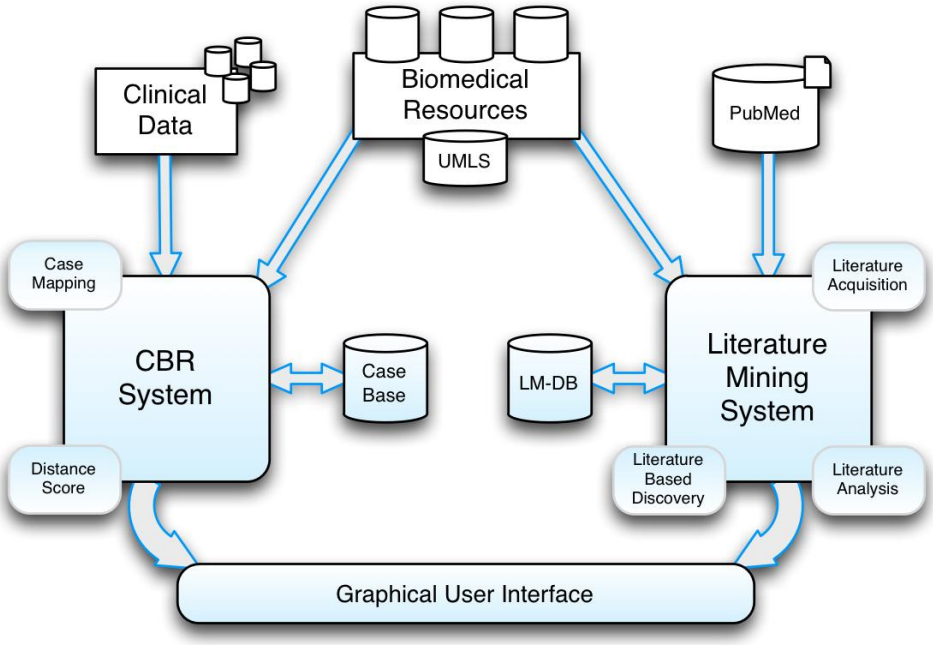

Figure 5.1: The developed system along with the exploited resources.

In particular, the present work is focused on knowledge management methodologies able to exploit both unstructured knowledge, available in the form of scientific literature, and structured knowledge related to real medical cases (typically contained in Electronic Medical Records, Electronic Health Records, clinical data warehouses, custom clinical databases and spreadsheets). The research effort has been divided in two main directions that led to the development of two sub-systems: the Literature Mining system and the Case Based Reasoning system. These two fields have been matter of methodological research conducted alongside with the actual implementation of a software system that puts into practice the studied techniques. Moreover, also the possible integration of such methods has been taken into consideration and investigated.

An outline of the overall system architecture is shown in Figure 5.1 where the two sub-systems are depicted along with the resources they exploit to perform their tasks. The system uses both purposely-developed resources and public biomedical resources, such as terminologies and scientific literature repositories.

The Literature Mining system's main goal is to manage efficiently the overwhelming amount of information coming from the scientific literature published every day all around the world; to achieve this goal it combines text mining technologies (in particular, Information Extraction techniques) with a persistence layer to make the extracted data easily and quickly available. The typical usages of the Literature Mining system are:

- summarization of a given set of articles through the biomedical concepts they contain (typically concepts cited in the article's abstract, but also its MeSH terms); the concepts (e.g. genes and diseases), once extracted from the source articles, are maintained in a purposely developed database for further access;
- retrieval of the articles concerning a set of biomedical concepts specified by the user; the literature query tool provided by our system is more flexible than the ones made available by the online repositories of scientific literature (PubMed, in particular); in fact, the query environment developed during my PhD allows the definition of selection criteria that deeply exploit biomedical terminologies (UMLS in particular) and the relations among concepts.
- discovery of new knowledge supported by indirect associations reported in literature; this operation is performed by the Literature Based Discovery system that exploits the Literature Mining system and the results of its automatic analysis of scientific papers.

The Case Based Reasoning system allows the comparison of patients coming from different databases where different terms can be used to describe their features set; this heterogeneity is managed by founding both the internal representation of patients' features and the distance

computation algorithm on UMLS. The Case Based Reasoning system is useful to:

- given an incoming patient described by a set of features, provide the cases that prove to be more similar to it according to the distance algorithm; this operation is performed by computing the distance of the new case from all the patients belonging to the base of known already-solved medical cases.
- produce a distance matrix between a set of known cases, by comparing each case with all the others, in order to identify clusters of cases that present similar features.

This chapter is structured as follows: Section 5.1 presents the basic technologies adopted to develop the whole system, Section 5.2 and Section 5.3 respectively describe the Literature Mining system and the Case Based Reasoning system implementations; finally, in Section 5.4 the developed Graphical User Interface, providing an easy access to some functionalities of the system, is shown.

5.1. Technologies

This section contains a detailed dissertation on the main technologies adopted to implement the Literature Mining system, the Case Based reasoning system and the Graphical User Interface.

5.1.1. The Unified Medical Language System

The Unified Medical Language System (UMLS) [152] is a collection of biomedical vocabularies linked to each other thanks to a common codification system onto which entries from different sources are mapped. The final goal of such an effort is to connect different vocabularies in order to obtain the best overall description of the medical knowledge.

Furthermore, UMLS provides also a semantic categorization of the contained concepts (i.e. each concept is associated to the semantic classes which it belongs to) and a network of semantic relationships between the classes.

UMLS, that comprises the UMLS Knowledge Sources (UMLSKS) (databases) and a set of software tools, was produced and is still maintained by the U.S. National Library of Medicine (NLM) with the purpose of facilitating the development of computer systems dealing with the comprehension of the language of biomedicine and health; in particular, the typical usage of the UMLS technologies regards the process of creating, processing, retrieving, and integrating biomedical and health data and information. Moreover, the software tools associated with UMLS help the

developers in customizing and using properly the UMLS Knowledge Sources.

Three different resources compose the UMLS Knowledge Sources: the Metathesaurus, the Semantic Network and the SPECIALIST Lexicon.

5.1.1.1. The UMLS Metathesaurus

The Metathesaurus is a large, multi-purpose, vocabulary database; the core elements of the Metathesaurus are the concepts, that belong to the biomedical and health related area, which names and relations with each other are stored in the database. In other words, the Metathesaurus links alternative names and views of the same concept and identifies useful relationships between different concepts.

The Metathesaurus is built from the electronic versions of several medical thesauri and classifications (the *source vocabularies*) used in different areas of the whole health sciences scenario (e.g. patient care, health services billing, public health statistics, indexing and cataloging biomedical literature, basic, clinical, and health services research).

As an aggregating source of knowledge, the Metathesaurus inherits the scopes of its source vocabularies and, despite some concepts and relationships have been added by the NLM curators to boost the coherence of the whole system, essentially all the concepts of the Metathesaurus belong to at least one of its source vocabularies.

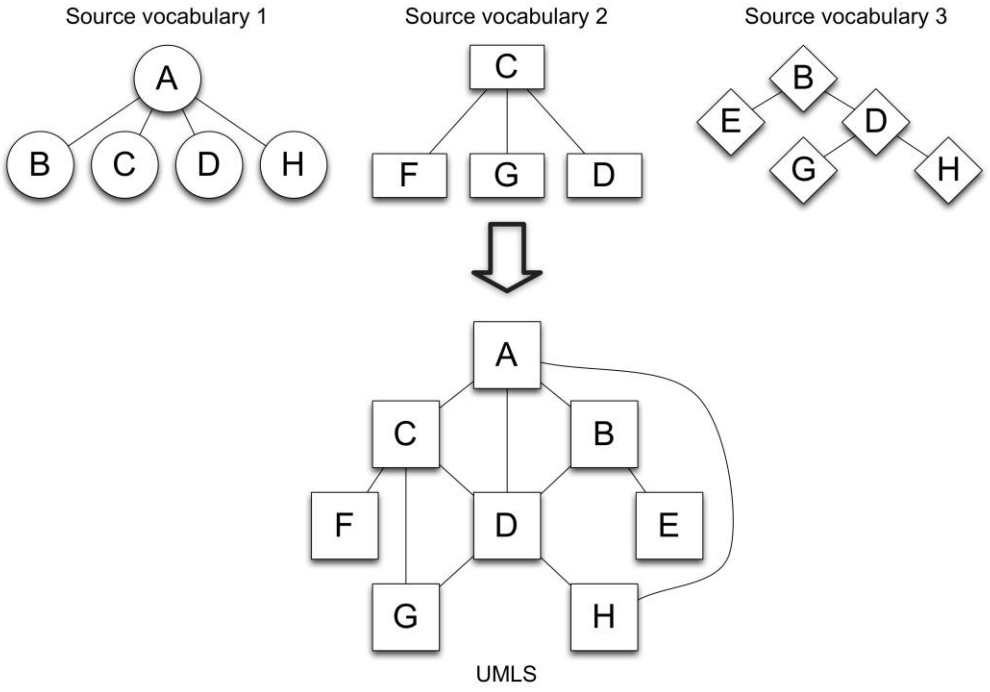

Figure 5.2: An exemplification of the building process of the UMLS meta-representation of health domain data.

The Metathesaurus preserves the meanings, concept names, and relationships from its source vocabularies also when the same term in two different vocabularies are associated with different concepts (i.e. different meanings) or when the same concept belongs to different hierarchies (i.e. has relationships with different families of concepts). This preservation priority is followed also when conflicting relationships between two concepts appear in different source vocabularies leaving to the user the accomplishment of a potential disambiguation.

In other words, as stated in [152]: "*the Metathesaurus does not represent a comprehensive NLM-authored ontology of biomedicine or a single consistent view of the world (except at the high level of the semantic types assigned to all its concepts). The Metathesaurus preserves the many views of the world present in its source vocabularies because these different views may be useful for different tasks*". In Figure 5.2 it is shown how the UMLS Metathesaurus exploits the information coming from different sources in order to build up its own meta-representation of health domain data.

Because of its multi-purpose nature, in order to be effectively exploited inside singular applications, the Metathesaurus needs to be customized; in particular, users have to select, among the available source vocabularies, which are the ones suitable to be integrated in the actual installation of the Metathesaurus that the system will use. This choice has significant effects on the operative usefulness of UMLS inside the system. Moreover, it is possible to exclude only particular subsets of terms, such as non-standard abbreviations, from the included source vocabularies.

The majority of source vocabularies are available under the standard (and quite open) Metathesaurus license; however, some vocabulary producers place additional restrictions on the use of their content as distributed within the Metathesaurus. To this day the source vocabularies available are more than 100. "*The Metathesaurus source vocabularies include terminologies designed for use in patient-record systems; large disease and procedure classifications used for statistical reporting and billing; more narrowly focused vocabularies used to record data related to psychiatry, nursing, medical devices, adverse drug reactions, etc.; disease and finding terminologies from expert diagnostic systems; and some thesauri used in information retrieval. A categorized list of the English-language source vocabularies is available*".

One of the primary purposes of the Metathesaurus is to connect to each concept all its different names reported in the source vocabularies; to this purpose it assigns to concepts several types of unique identifiers and their names without loosing trace of the original codification in the source vocabularies. In addition the Metathesaurus stores, for every concept name, additional information such as: language, vocabulary source and name type. The unique identifiers used within the Metathesaurus are:

- Concept Unique Identifier (CUI) – *"A concept is a meaning. A meaning can have many different names. A key goal of Metathesaurus*

construction is to understand the intended meaning of each name in each source vocabulary and to link all the names from all of the source vocabularies that mean the same thing (the synonyms)".

- CUI identifies univocally and permanently a concept within the Metathesaurus, therefore all the synonyms will have the same CUI. Generally CUIs don't change over time except for the case when it is discovered that two concepts refer to the same entity and they are merged together.
- String Unique Identifier (SUI) – SUIs are the unique identifiers for names assigned to concepts (strings, indeed). Each variation in the character set that composes the string (e.g. uppercase/lowercase) produces a different string with a different SUI. In the case of strings associable to different concepts (e.g. "tree" can refer to a way of representing a hierarchy and also to a perennial woody plant) they will maintain a single SUI that will be associated with more CUIs.
- Atoms Unique Identifier (AUI) – AUIs map the single occurrence of a string in a source vocabulary. Therefore, if the same string is present in more vocabularies, each occurrence will be assigned with a different AUI, all these atoms will be linked to the same SUI and to a single CUI (potentially not the same for all the AUIs), since, within a single vocabulary, a single occurrence of a term can have only one meaning.
- Lexical Unique Identifier (LUI) – *"For English language entries in the Metathesaurus only, each string is linked to all of its lexical variants or minor variations by means of a common term identifier (LUI). […] Like a string identifier, the LUI for an English string may be linked to more than one concept. This occurs when strings that are lexical variants of each other have different meanings. In contrast, each string identifier and each atom identifier can only be linked to a single LUI".*

In Table 5.1 it is shown an example of how this unique identifiers are used together.

Each concept in the Metathesaurus is also characterized by at least one *Semantic Type* (codified with a Type Unique Identifier - TUI). This association is done by the Metathesaurus curators at NLM. An example of concept/TUI associations is shown in Table 5.2.

Beyond the synonyms relationships that are naturally managed by the Metathesaurus for its own nature, many other relationships between concepts are also included; the vast majority of relationships come from the source vocabularies, some have been added by the curators at NLM and others have been proposed by the community of UMLS users in order to adapt the system to certain scopes.

The Metathesaurus contains two types of non-synonymous relationships: intra-source vocabulary and inter-source vocabulary relationships. The first are relationships between concepts that come from the same source vocabulary, while the second connect concepts belonging to different

vocabularies. It is important to notice that "*the Metathesaurus does not include all possible non-synonymous relationships between the concepts it contains*", but only relationships coming from the source vocabularies and some others added in order to connect related concepts.

Table 5.1: An example of the several unique identifiers of the Metathesaurus used together.

Concept (CUI)	Terms (LUIs)	String (SUIs)	Atoms (AUIs)
C0004238 Atrial Fibrillation (preferred) Atrial Fibrillations Auricular Fibrillation Auricular Fibrillations	**L0004238** Atrial Fibrillation (preferred) Atrial Fibrillations	**S0016668** Atrial Fibrillation (preferred)	**A0027665** Atrial Fibrillation (from MSH) **A0027667** Atrial Fibrillation (from PSY)
		S0016669 (plural variant) Atrial Fibrillations	**A0027668** Atrial Fibrillations (from MSH)
	L0004327 (synonym) Auricular Fibrillation Auricular Fibrillations	**S0016899** Auricular Fibrillation (preferred)	**A0027930** Auricular Fibrillation (from PSY)
		S0016900 (plural variant) Auricular Fibrillations	**A0027932** Auricular fibrillations (from MSH)

Table 5.2: Associations between a single concept (C0001771) and its semantic types.

CUI	Name	Semantic Type	TUI
C0001771	Agar	Carbohydrate	T118
C0001771	Agar	Pharmacologic Substance	T121
C0001771	Agar	Indicator, Reagent, or Diagnostic Aid	T130

Some examples of intra-source relationships are "parent", "child" and "sibling"; all these relationships are "distance-1" hierarchical relationships (i.e. they connect two concepts directly). Other types of intra-source relationships are statistical relationships; these relationships connect two concepts from the same vocabulary when they co-occur in a specific context, such as co-occurrences in literature or in discharge summaries. In contrast to standard intra-source relationships, statistical relationships can connect very different concepts, such as diseases and drugs.

The primary inter-source relationships are the synonyms that, thanks to the structure of the Metathesaurus, join under the same concept also terms coming from different source vocabularies; for example some "orphan" concepts (with few or no ancestors, siblings, or children in their own source vocabularies) have been manually connected to richer contexts in other vocabularies and also some "like" or "similar" relationships have been added in order to integrate to the strict view of synonymy.

5.1.1.2. The UMLS Semantic Network

The second UMLS knowledge source available is the Semantic Network, that consists in a set of Semantic Types, namely subject categories which the concepts in the Metathesaurus belong to, and a set of Semantic Relations that represent the possible relationships that can connect two Semantic Types. The Network contains 133 Semantic Types and 54 relationships.

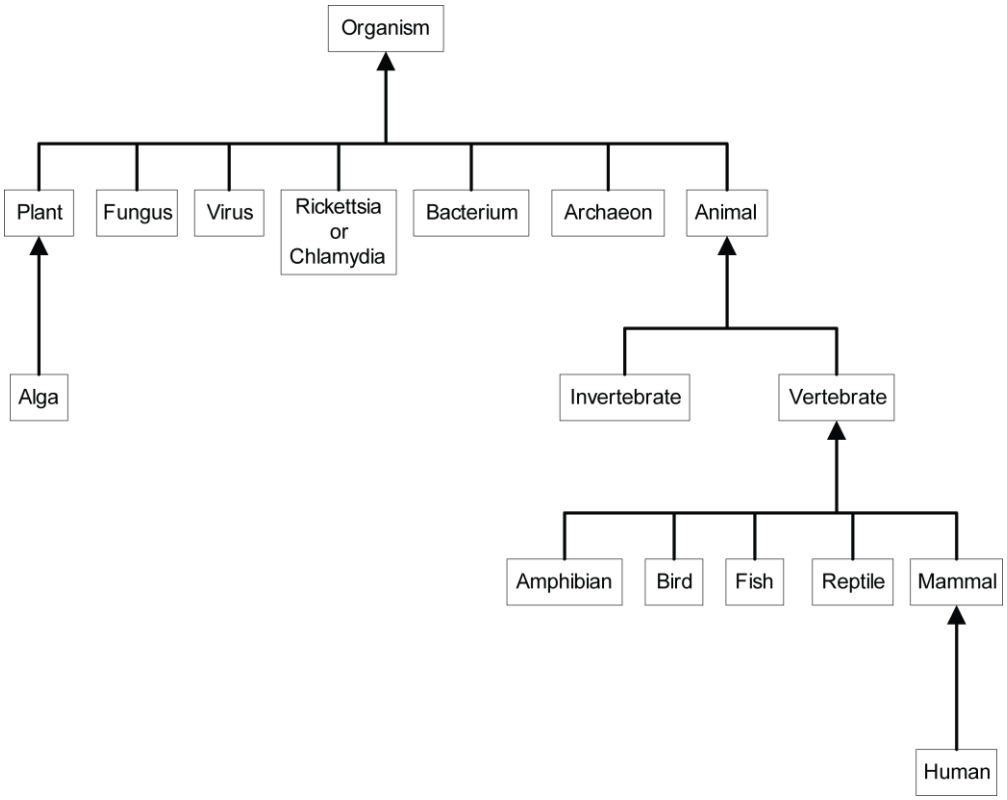

Figure 5.3: The "Organism" major grouping of the Semantic Network. The connections between the Semantic Types represent "is a" relationships.

Semantic Types represent the nodes of the Network while relationships represent the link between the nodes. The nodes are further grouped in 7 major groupings that are:

- Organisms
- Anatomical Structures
- Biologic Function
- Chemicals
- Events
- Physical Objects
- Concepts or Ideas.

All the concepts in the Metathesaurus are assigned to at least one Semantic Type.

In Figure 5.3 it is represented a part of the Semantic Network: the "Organism" major grouping. The links shown in figure are the primary links of the Network and represent the basic relationship of all the hierarchies: the "is a" relation. This is a primary relation used when deciding the semantic type to be assigned to the single concept in the Metathesaurus. In general, it is chosen the most specific available semantic type. In addition to the "is a" relation, the Semantic Network is characterized by many other types of non-hierarchical relationships. The relationships themselves are grouped into a hierarchy which has 5 root relationships:

- "Physically related to"
- "Spatially related to"
- "Temporally related to"
- "Functionally related to"
- "Conceptually related to".

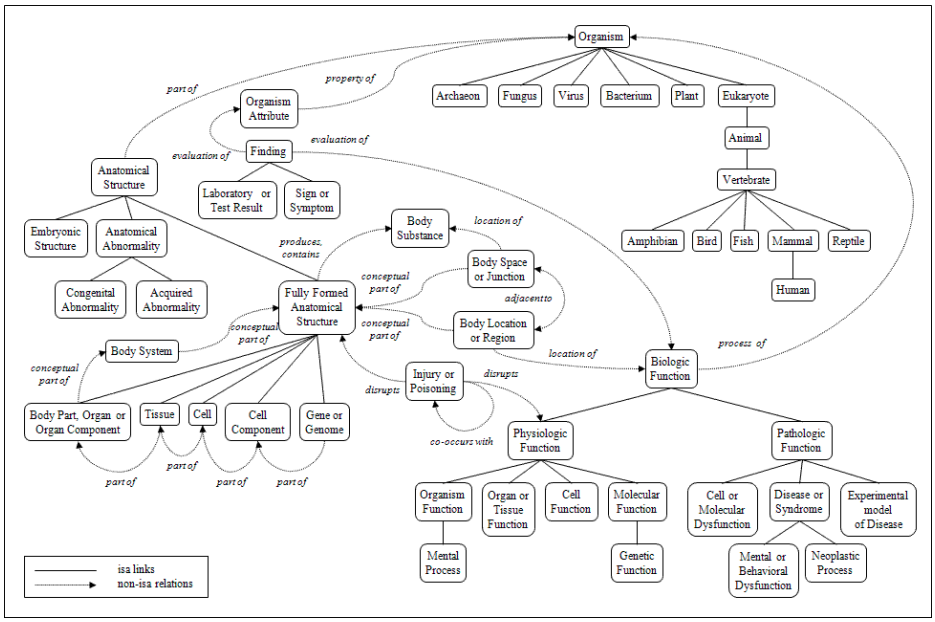

Figure 5.4: A detail of Semantic Network with hierarchical and non-hierarchical relationships.

Figure 5.4 shows a portion of the Semantic Network, illustrating relations, either hierarchical or not, that exist between the semantic types.

Relationships between Semantic Types are asserted at the higher possible level in the hierarchy and are inherited by the child types (i.e. connected with the "is a" hierarchical link); such relations are between Semantic Types and are not applied automatically to all concepts belonging to those types; furthermore, when the inherited relationship between two Semantic Types is illogical, it is blocked.

5.1.1.3. The UMLS SPECIALIST Lexicon

Finally, the SPECIALIST Lexicon is a vocabulary of biomedical terms and common use terms; it is intended to be a general lexicon integrated with biomedical terms. Each record of the lexicon comprehends syntactic, morphological and orthographic information. The SPECIALIST Lexicon is the primary data source for the SPECIALIST NLP tools provided in the UMLS.

5.1.2. The General Architecture for Text Engineering

The General Architecture for Text Engineering (GATE) [153] is "*a framework and graphical development environment which enables users to develop and deploy language engineering components and resources in a robust fashion. […] The framework can be used to develop applications and resources in multiple languages, based on its thorough Unicode support*".

In order to produce robust software that can be used to process human language, it is important to focus the development efforts on the engineering aspects of this software; in fact a well-developed NLP software must comply with several aspects and, in particular with predictability; GATE facilitates the design of NLP applications that respect the basic software engineering principles from several point of view:

- There is a clean separation between low level tasks, such as data storage, data visualization and loading, and high level components that manage the data structures and the algorithms that actually perform the language processing task.
- The framework provides tools to easily manage the performance measure of the NLP tasks performed.
- The use of XML as baseline standard for data representation and also for the communication between the system's components, reduces the integration overheads.
- Beyond the robust framework, GATE provides also a large set of language analysis components in form of APIs. These components accomplish the vast majority of basic NLP tasks and, therefore, are

usually exploited to build language analysis pipelines. Furthermore these components can be extended or replaced according to the specific needs of the user.

As architecture for text engineering (TE), GATE defines the general structure of a TE application; it assigns the responsibility to the components that form the application and establishes how they should interact with each other. As a framework it provides a reusable design for the TE systems and a set of pre-developed components that can be combined and integrated in order to build up the desired TE applications. As a development environment it allows the user to minimize the time spent in writing code and debugging applications, thanks to development-aiding and debugging tools.

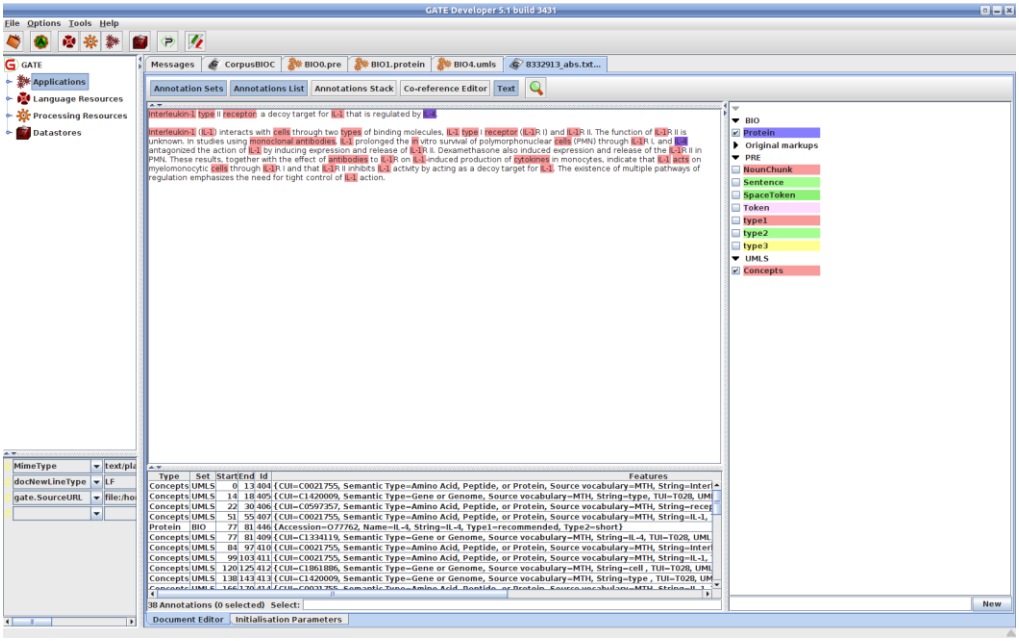

Figure 5.5: A screenshot of GATE's Graphical User Interface.

The main element of the architecture are the components, therefore GATE is usually defined as a component-based architecture. Components can be easily added or removed from the applications that, ultimately, can be represented as pipelines where several components are put in cascade and executed sequentially. This ease in managing the TE applications developed with GATE allows an effective and simple comparison of different implementations, for instance by adding or removing components from the pipeline or to replace a component performing a specific task with other components absolving the same task with different algorithms.

The framework is composed by a core library and a set of reusable modules (i.e. the components or plugins). The core library defines the general architecture and provides a set of facilities to visualize data and

easily manage the input/output operations. The modules typically perform the basic tasks of TE applications (e.g. tokenization, part-of-speech tagging) and, therefore, allow the developers to focus on most specific components of their system without spending efforts in re-developing the basic components.

The framework provides also a Graphical User Interface (GUI). Within this GUI all the applications developed can be executed and tested and the developer has an immediate feedback on their actual performance. In Figure 5.5 it is a screenshot of GATE's GUI. Moreover, applications developed within the framework can be executed also outside the GUI thanks to the GATE Java APIs in order to build up standalone applications, which can run independently of the development environment. Finally, developers can design and implement custom modules in order to add to their systems new LE features tailored on their specific needs.

GATE manages data, algorithms and ways of visualizing them with so many categories of components or resources:

- *Language Resources* represent the data inside the framework. These data can be, for instance, document to analyze, corpora (set of documents), lexicons and ontologies.
- *Processing Resources* represent algorithms that execute several LE tasks, such as tokenization and parsing of texts. Processing resources typically work on some language resources (i.e. corpora and documents) exploiting the data of other language resources (e.g. lexicons and ontologies).
- *Visual Resources* describe the visual components that are used within the GUI.

The separation between the three types of resources allows an independent development of data and algorithms and also the use of different views on the same data.

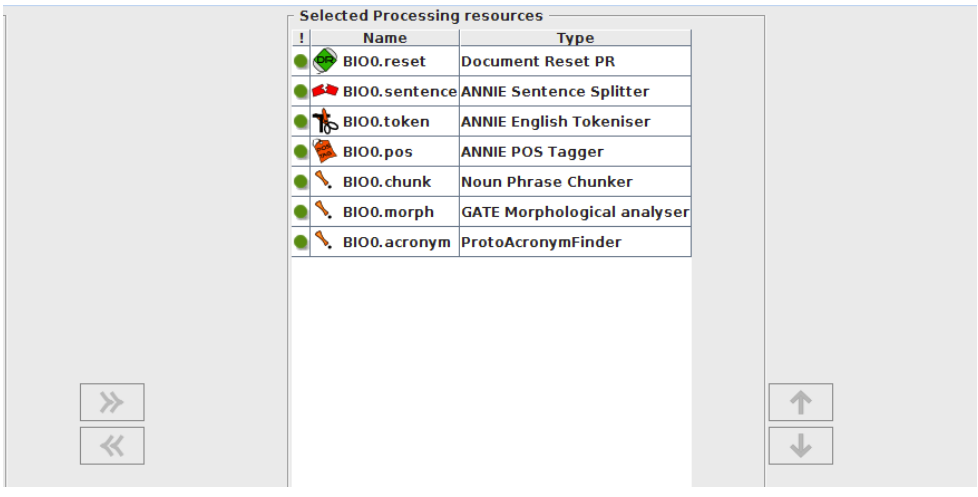

Figure 5.6: The pipeline definition tool of the GATE's GUI.

In general, the resources are known as CREOLE (a Collection of Reusable Objects for Language Engineering); CREOLE are declared in a XML file which defines their name, implementing class, parameters etc. The framework uses these description files in order to discover the available resources and regulate their combination in complex applications.

A set of processing resources combined to work together in a sequential way is called pipeline; in GATE it is possible to develop several types of pipelines, the most used are *Simple Pipelines*, that work on a single document, and *Corpus Pipelines* that recursively execute the processing resources on all the documents belonging to a corpus. For both the types of pipelines, the framework provides also a "conditional" version that allows the user to set up some decision points within the execution of the processing resources in order to change the execution flow on the basis of some conditions. In Figure 5.6 it is shown how users can build up pipelines, within the GUI, by combining several processing resources.

When a document is created or opened in GATE, the framework, on the basis of its extension, the system performs its conversion into a single unified model of annotation that ensures a smooth communication between components. This model is essentially based on feature/value pairs; features are Java *String* objects, while the associated values can be any type of Java objects. In practice, each annotated document is composed by its own content (plain text with a set of position indicators called "*nodes*" acting as bookmarks inside the text) and by an arbitrary number of *Annotation Sets* (structures that store the association between parts of the text with the relative feature/value pairs, called *annotations)* produced by the processing resources executed on the document. In order to identify which part of the text is associated with the single annotation, the model exploits the nodes that bookmark the document. The structure of an annotated sample document represented in the internal format of the framework is shown below:

```
<?xml version='1.0' encoding='UTF-8'?>
<GateDocument>

<!-- The document's features-->
<GateDocumentFeatures>
 […]
</GateDocumentFeatures>

<!-- The document content area with serialized nodes -->
<TextWithNodes>
<Node id="0" />Dilated<Node id="7" /> <Node id="8"
/>cardiomyopathy<Node id="22" /> <Node id="23" />or<Node
id="25" /> <Node id="26" />DCM<Node id="29" /> <Node
id="30" />is<Node id="32" /> <Node id="33" />a<Node
id="34" /> <Node id="35" />condition<Node id="44" />
<Node id="45" />in<Node id="47" /> <Node id="48"
/>which<Node id="53" />
<Node id="211" />
```

```
[…]
</TextWithNodes>

[…]

<!-- Named annotation set -->
<AnnotationSet Name="PRE">
 <Annotation Id="2" Type="Token" StartNode="0"
EndNode="7">
       <Feature>
                   <Name
className="java.lang.String">length</Name>
                   <Value
className="java.lang.String">7</Value>
       </Feature>
       <Feature>
              <Name
className="java.lang.String">category</Name>
              <Value
className="java.lang.String">NNP</Value>
       </Feature>
       <Feature>
             <Name
className="java.lang.String">orth</Name>
             <Value
className="java.lang.String">upperInitial</Value>
       </Feature>
       <Feature>
             <Name
className="java.lang.String">root</Name>
             <Value
className="java.lang.String">dilated</Value>
       </Feature>
       <Feature>
             <Name
className="java.lang.String">kind</Name>
             <Value
className="java.lang.String">word</Value>
       </Feature>
       <Feature>
             <Name
className="java.lang.String">string</Name>
             <Value
className="java.lang.String">Dilated</Value>
       </Feature>
 </Annotation>

[…]
```

In this example the header of the document, the bookmark nodes and a single annotation performed by a tokenizer and a POS tagger are shown.

GATE offers also several storage mechanisms for the analyzed data: a mechanism based on relational databases and two mechanisms based on

file systems that, respectively, exploit Java serialization and the internal XML-based format.

Some of the reusable processing resources provided with GATE are packaged together to form ANNIE (A Nearly New Information Extraction system). Among the most used ANNIE's processing resources we find:

- The *ANNIE English Tokenizer*, that works on the document in two steps: the first is the identification of parts of text separated by spaces, the second deals with treating the exceptions to the simple rules of tokenization typical of English (e.g. words separated by a dash or an apostrophe).
- The *ANNIE Sentence Splitter*, that separates the phrases in the text, distinguishing points that represent the end of the sentence by others (such as those used for abbreviations).
- The *ANNIE POS Tagger*, that associates each element identified by the tokenizer with the corresponding grammatical class (e.g. noun, verb, adjective). The POS Tagger is a modified version of the one described in [154].

Another processing resources that is provided with the framework outside ANNIE and has been used within this work is:

- The *Noun Phrase Chunker*, that aims to identify inside the text specific structures, called *noun phrases*, subsets of consecutive words that, taken together, represent a nominal entity and that, in association with *verb phrases*, compose the sentences in the text. The chunking algorithm of the Noun Phrase Chunker is based on the work described in [55]. The noun phrases extraction process is very important since biomedical concepts are often represented by more than one word and letting the information extraction components work on noun phrases instead of complete sentences is more efficient.
- The *GATE Morphological Analyzer*, that, considering one token and its part of speech tag identifies its lemma and an affix working with certain regular expression rules. These values are than added as features on the Token annotation.

5.1.3. The Entrez Utilities

Entrez [155] is a search system developed by the National Center for Biotechnology Information (NCBI) in the U.S.; this search system is a global cross-database query system that allows to simultaneously access PubMed (the primary access point to MEDLINE) and other 38 literature and molecular databases including DNA and protein sequence, structure, gene, genome, genetic variation and gene expression. A non-exhaustive list of the databases that compose Entrez, to date, is shown in Table 5.3. The

standard way to access Entrez is the web interface available at (http://www.ncbi.nlm.nih.gov/gquery).

Table 5.3: A non-exhaustive list of the databases that compose Entrez.

Database name	Description
BioSystems	The BioSystems database collects information on interacting sets of biomolecules involved in metabolic and signaling pathways, disease states, and other biological processes.
Bookshelf	The NCBI Bookshelf contains a collection of full-text books that can be searched online and that are linked to PubMed records through research paper citations within the text.
Conserved Domains	Conserved Domains is a database of protein domains represented by sequence alignments and profiles for protein domains conserved in molecular evolution.
dbGaP	dbGaP (Database of Genotypes and Phenotypes) provides the results of studies that have investigated the interaction of genotype and phenotype.
dbVAR	dbVAR (Database of Genomic Structural Variation) contains information about large-scale genomic variation.
Epigenomics	The Epigenomics database contains results of genome-wide studies on modifications of chromatin (histone modification, DNA methylation, DNAase footprinting) in various cell types that assay programmable changes that affect gene expression (epigenetics).
EST	The EST database contains sequence records from the bulk EST (Expressed Sequence Tag) division of GenBank.
Gene	Gene is a searchable database of genes, focusing on genomes that have been completely sequenced and that have an active research community to contribute gene-specific data.
Genome	The Genome database contains sequence and map data from the whole genomes of over 1000 species or strains.
Genome Project	Genome Projects collects information on complete and in-progress large-scale sequencing, assembly, annotation, and mapping projects for cellular organisms.
GEO Datasets	GEO Datasets stores curated gene expression and molecular abundance data sets assembled by NCBI from the Gene Expression Omnibus (GEO) repository of microarray data.
GEO Profiles	GEO Profiles is a database that stores individual gene expression and molecular abundance profiles assembled from the Gene Expression Omnibus (GEO) repository of microarray data.
GSS	The GSS database contains sequence records from the bulk GSS (Genome Survey Sequence) division of GenBank.
HomoloGene	The HomoloGene database contains automatically generated sets of homologous genes and their corresponding mRNA, genomic, and protein sequence data from selected eukaryotic organisms.
MeSH	MeSH (Medical Subject Headings) is the National Library of Medicine's controlled vocabulary and classification system (ontology) used for indexing articles in PubMed. MeSH terminology provides a consistent way to retrieve information that may use different terminology for the same concepts.
NCBI Web Site Search	NCBI Site Search is database of static NCBI web pages, documentation, and online tools.

NLM Catalog	The NLM Catalog contains records for books, journals, audiovisuals, computer software, electronic resources, and other materials in the National Library of Medicine (NLM) collections.
Nucleotide	Apart from sequence data in the EST (Expressed Sequence Tag) and GSS (Genome Survey Sequence divisions of GenBank, the Nucleotide database contains all the sequence data from GenBank, EMBL, and DDBJ, the members of the International Nucleotide Sequence Databases Collaboration (INSDC).
OMIA	OMIA (Online Mendelian Inheritance in Animals) is a database of genes, inherited disorders and traits in animal species (other than human and mouse).
OMIM	The OMIM (Online Mendelian Inheritance in Man) database contains review articles human genes, genetic disorders, and other inherited traits. OMIM articles provide links to associated literature references, sequence records, maps, and related databases.
PopSet	The PopSet database contains related nucleotide sequences that originate from comparative studies: phylogenetic, population, environmental (ecosystem), and mutational.
Probe	Probe is a database of nucleic acid reagents designed for use in a wide variety of biomedical research applications including genotyping, gene expression studies, SNP discovery, genome mapping, and gene silencing.
Protein	The Protein database contains amino acid sequences created from the translations of coding regions provided on nucleotide records in GenBank, EMBL, and DDBJ, the members of the International Nucleotide Sequence Databases Collaboration (INSDC) as well as those from coding regions on NCBI Reference Sequences and the Third Party Annotation (TPA) database records.
Protein Clusters	Protein Clusters is a collection of related protein sequences (clusters) consisting of Reference Sequence proteins that are encoded by complete prokaryotic genomes as well those encoded eukaryotic organelle plasmids and genomes.
PubChem BioAssay	PubChem BioAssay is a database that contains bioactivity screens of chemical substances described in PubChem Substance.
PubChem Compound	The PubChem Compound database contains unique, validated chemical structures (small molecules) that can be searched using names, synonyms or keywords.
PubChem Substance	The PubChem Substance database contains information on chemical substances including mixtures electronically submitted to PubChem by depositors
PubMed	PubMed is database of citations and abstracts for biomedical literature from MEDLINE and additional life science journals. Links are provided when full text versions of the articles are available through PubMed Central or other websites.
PubMed Central	PubMed Central (PMC) is the U.S. National Library of Medicine's digital archive of life sciences journal literature. PMC contains full-text manuscripts deposited by authors or articles provided by the publisher.
SNP	The SNP (Single Nucleotide Polymorphism) database is a central repository for single nucleotide polymorphisms, microsatellites, and small-scale insertions and deletions.

SRA	The SRA (Sequence Read Archive) contains sequencing data from the next generation sequencing platforms.
Structure	The Structure or Molecular Modeling Database (MMDB) contains experimental data from crystallographic and NMR structure determinations.
Taxonomy	The Taxonomy database contains the names and phylogenetic lineages of the more than 160,000 organisms that have molecular data in the NCBI databases.
UniGene	UniGene is a database that provides automatically generated nonredundant sets (clusters) of transcript sequences, each cluster representing a distinct transcription locus (gene or expressed pseudogene).
UniSTS	UniSTS is a comprehensive database of sequence tagged sites (STSs) derived from STS-based maps and other experiments.

The Entrez Programming Utilities (E-Utilities) are a set of server-side services that provide a stable access point to the Entrez search system. E-Utilities can be accessed in two ways: with an HTTP interface, with a fixed URL syntax, and via web service, using the Simple Object Access Protocol (SOAP). The fixed URL syntax translates the input parameters that define the Entrez query into an URL, thus the system can acquire them in order to exploit the various NCBI software components to search for and retrieve the requested data. The typical client-server interaction with this fixed URL syntax is the following:

- The client posts to NCBI an E-Utility URL that describes the query to perform.
- On the server side at NCBI, the Entrez system parses the URL to extract the query parameters, performs the user-defined query and produces the response in XML format.
- The server sends back to the client the XML response that contains the results produced by the query.
- On the client side the XML response is parsed in order to make the query results available.

This architecture allows the client to be implemented in many software languages, as long as they provide functionalities to send an URL to the E-Utilities and to interpret the XML response; examples of such languages are Perl, Python, C++ and Java.

As an alternative to the fixed URL syntax access, the NCBI makes a web service based on the Simple Object Access Protocol (SOAP) available. Besides this alternative access mode the NCBI has also developed software libraries that allow an easy integration of this web service within the Java runtime environment [156].

Within the Entrez system each record of the 39 databases is identified with an integer value called UID (unique identifier); the E-utilities use UIDs for both data input and output.

E-Utilities are composed by a set of eight server-side programs, each one suited to accomplish a specific querying task on Entrez [157]:

- *EInfo - Provides the number of records indexed in each field of a given database, the date of the last update of the database, and the available links from the database to other Entrez databases.*
- *ESearch - Responds to a text query with the list of matching UIDs in a given database (for later use in ESummary, EFetch or ELink), along with the term translations of the query.*
- *EPost - Accepts a list of UIDs from a given database, stores the set on the History Server, and responds with a query key and web environment for the uploaded dataset.*
- *ESummary - Responds to a list of UIDs from a given database with the corresponding document summaries.*
- *EFetch - Responds to a list of UIDs in a given database with the corresponding data records in a specified format.*
- *ELink - Responds to a list of UIDs in a given database with either a list of related UIDs (and relevancy scores) in the same database or a list of linked UIDs in another Entrez database; checks for the existence of a specified link from a list of one or more UIDs; creates a hyperlink to the primary LinkOut provider for a specific UID and database, or lists LinkOut URLs and attributes for multiple UIDs.*
- *EGQuery - Responds to a text query with the number of records matching the query in each Entrez database.*
- *ESpell - Retrieves spelling suggestions for a text query in a given database.*

For the purposes of the present work we have exploited two of these eight services: ESearch and EFetch. ESearch executes the following functions:

- *Provides a list of UIDs matching a text query*
- *Posts the results of a search on the History server*
- *Downloads all UIDs from a dataset stored on the History server*
- *Combines or limits UID datasets stored on the History server*
- *Sorts sets of UIDs*

In order to set up a query with ESearch users have to define several parameters, some are required, but the majority is optional and users can choose to specify them in order to tailor the query on their specific needs. The requested parameters are:

- "*db*", whose value must be the valid Entrez database name on which the query is performed;
- "*term*", that represent the query to perform.

The ESearch optional query parameters are:

- "*usehistory*", when set to "*y*" (yes), ESearch posts the query results to a History server in order to use them in a subsequent E-utility call.
- "*WebEnv*", identifies the context of the global search operation performed through several E-Utilities. When the *WebEnv* is used, ESearch will post the results of the search operation to this pre-existing *WebEnv*, thereby appending the results to the existing environment.
- "*query_key*", is an integer identification number returned by a previous search operation performed with an E-utility; when *query-key* is specified, ESearch filters the resulting UIDs of the query specified by *term* with those of the previous query identified by *query-key* with a logical-AND operation.
- "*retstart*", since the result of the ESearch querying process is a set of UIDs, it is possible, setting a specific integer *retstart*, to obtain a subset of these results that starts from the index identified by *retstart*.
- "*retmax*", is the maximum number of UIDs that are included in the XML output of ESearch. The default value (i.e. when *retmax* is not specified) is 20. This value does not interfere whit the history-storing mode that is managed with *usehistory*, *WebEnv* and *query_key*, but exclusively with the direct XML output of ESearch.
- "*rettype*", defines the return type of ESearch. There are two possible return values: the default fashion is a list of UIDs but it is also possible to request only an integer value representing the global number of results.
- "*field*", if specified the ESearch query term will be searched only in a specific field of the Entrez entry. For instance, in the case of PubMed searches, it is possible to limit the research in the title, author list or abstract of the entry (i.e. an article).
- "*datetype*", since several dates can be associated whit an Entrez entry, it is possible to specify on which dates the ESearch utility should apply the filters specified by *reldate*, *mindate* and *maxdate*.
- "*reldate*", when set to an integer *n*, ESearch returns only the results that have the date specified by *datetype* in the last *n* days.
- "*mindate*" and "*maxdate*", are used in order to specify the date range to which the retrieved entries must belong. The type of date used is the one specified by *datetype*.

The other E-Utility used in this work is EFetch, which functions are:

- Returns formatted data records for a list of input UIDs
- Returns formatted data records for a set of UIDs stored on the Entrez History server

Therefore EFetch is the utility that, provided with a set of UIDs (both directly and with the history mode), actually returns the data that are

identified by each UID. These data depend on the nature of the entry retrieved within the entire Entrez system and, in the case of a PubMed query, contain all the information relative to a single article, like for example:

- The article's identification code within PubMed.
- The title of the article.
- The abstract of the article.
- The creation date of the article.
- The ISSN code of the journal where the article is published.
- The name of the journal.
- The list of the authors' names.
- The mesh terms that describe the article in PubMed.

Also for EFetch users have to define the querying parameters, some required and some optional. The required parameters are:

- "*db*", as for ESearch, identifies the Entrez database on which EFetch will run its query.
- "*id*", (used when the UIDs to fetch are passed directly) identifies the entries on which the EFetch tool is applied.
- "*query_key*" and "*WebEnv*", (used when EFetch acquires the UIDs from the History server, as described for ESearch) also identify the entries on which EFetch will work.

The EFetch optional parameters are:

- "*retmode*", the return mode parameter specifies the output format of the EFetch utility: plain text, HTML or XML.
- "*rettype*", specifies the return type; for instance, in the case of a PubMed query, two return types are available: "Abstract" or "MEDLINE".
- "*retstart*", as for ESearch, identifies the starting index for the return list.
- "*retmax*", as for ESearch, this parameter set the maximum number of entries included in the output of the utility. The maximum value of *retmax* allowed is 10000; in order to deal with output sets larger than this value, it is possible to iterate the EFetch procedure exploiting the *retstart* parameter.

5.1.4. The Google Web Toolkit

The Google Web Toolkit (GWT) is a set of open source tools that provide the developers with the resources to implement complex web-applications with the Java language; for the purposes of the present work we have used GWT to develop the Graphical User Interface (GUI) of the developed system.

The need to build web applications results from the many advantages that Rich Internet Applications (RIAs) have respect classic desktop applications: they do not require to be installed on the system where they are executed (a simple web browser in the client system is the only requirement in the vast majority of cases), the updates to the application are automatic and involve only the server-side and these applications are, in general, more scalable because most of the computational work is executed by the server, while the browser (on the client-side) is involved only in visualization tasks [158].

The AJAX (Asynchronous JavaScript and XML) technology allows web applications to be developed on the basis of a background data exchange between the server and the web browser, so that dynamic parts of the application pages can be updated without the complete reload of the whole web page, resulting in a seamless interaction experience similar to the one allowed by standard desktop applications. The other side of this coin is represented by the fact that AJAX applications strongly rely on JavaScript that makes their development particularly challenging; in fact, JavaScript is a complex language that requires a lot of practice in order to be exploited to create efficient applications and, furthermore, different web browsers frequently interpret the JavaScript code in different ways, making an uniform development and the debug process two hard challenges [159].

GWT was developed to overcome these problems by providing an abstraction layer that hides the JavaScript code and automatically evens the differences between the web browsers. GWT has been released by Google in the 2006 under the Apache License [160]. Beyond the automatic generation and optimization of JavaScript code, other advantages of using GWT are the reusability of the code, the ability to create dynamic web pages exploiting AJAX asynchronous calls, the easiness to internationalize the implemented application and the guaranteed portability between several web browser [159].

The GWT environment is composed by four main elements:

- a Java-to-JavaScript compiler that translates the developed Java code into the executed JavaScript code.
- JSNI (JavaScript Native Interface) allows integrating JavaScript code directly into the application's Java source code. This can be useful when the developer needs to integrate inside the application some handwritten JavaScript code, third-party JavaScript libraries or when he needs to access some low-level browser functionalities that are not exposed by the GWT APIs.
- JRE emulation library contains the Java-to-JavaScript translation of several libraries belonging to the Java Runtime Environment

(JRE). This library is the core of the automatic JavaScript code generation from the developed Java code.

- GWT APIs are a set of libraries that are used directly from the Java environment in order to create the web application; inside these APIs the graphical libraries used to create the GUIs, a graphical environment that allows to directly create these GUIs following the "What You See Is What You Get" principle and the Remote Procedure Call (RPC) libraries that manage the communication between client and server are included.

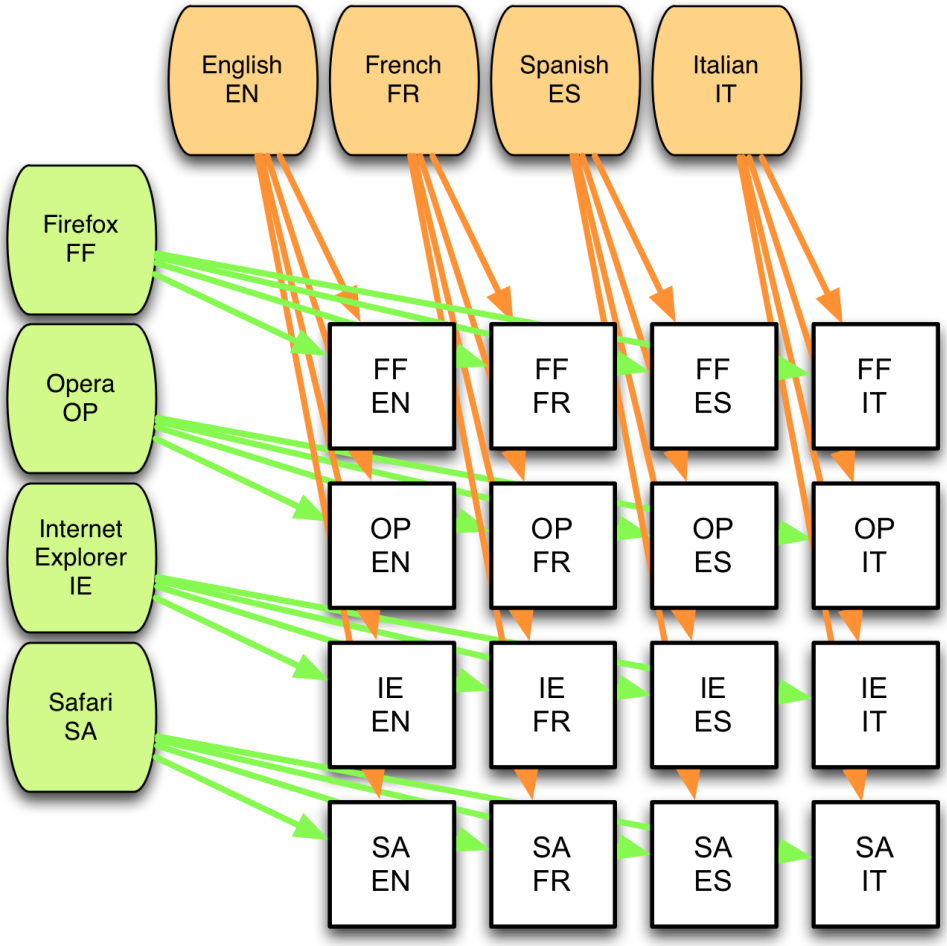

Figure 5.7: The multiple compilation process of GWT.

The core of GWT environment is the combination of the Java-to-JavaScript compiler and the JRE emulation library (eventually with the addition of JSNI) [161] that produces code that can be run on the vast majority of web browsers, such as Mozilla Firefox, Google Chrome, Microsoft Explorer, Apple Safari and their mobile versions. The operation of translating Java code into JavaScript has been highly optimized, thus only the actually executed code is translated and, when the code is compiled, GWT generates

many version of it, each one optimized for the specific features of the single web browsers supported. In addition this multiple compilation process is multiplied for every local version of the application implemented. In Figure 5.7 it is exemplified the multiple compilation process.

For the purposes of the present work, among the available GWT APIs, we have used those addressed to the development of the graphical interface of the system and those that manage the client-server interaction.

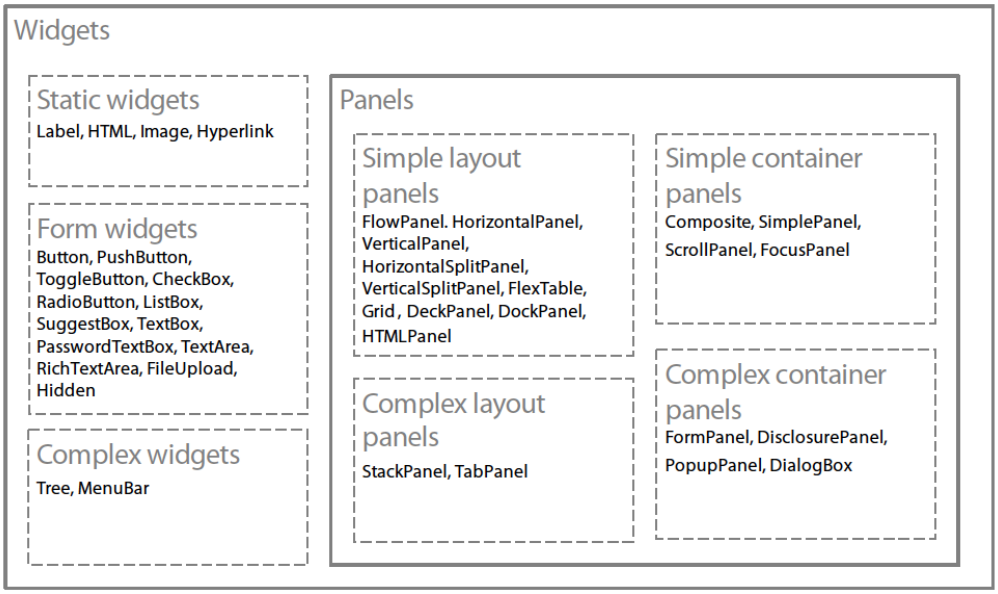

Figure 5.8: A taxonomy of GWT Widgets on the basis of their functionalities.

The graphical libraries (*com.google.gwt.user.client.ui*) allow the developer to create dynamic user interfaces by combining together the single components that compose the GUI. These components are called *Widgets* and range from simple ones (like buttons or labels) to more complex ones that allow representing very complex views. Some of the widgets are directly connected with specific HTML elements, while others are constituted by a combination of many basic HTML elements and can manage adequately the interactions that users have with these elements. It is possible to group the different GWT widgets on the basis of the functionalities they provide [162], see Figure 5.8:

- *Static Widgets* are the simplest type of GWT widgets; these components do not have an internal state and do not change dynamically on their own. When they are part of the user interface, the system can change their properties or their location but they should not change as a result of a user action. Examples of this type of widgets are *Label* and *Image*.

- *Form Widgets* are typically used inside HTML forms. The standard way for using forms in plain HTML is to submit the form to refresh the page with the results. In GWT (and generally in AJAX applications) the page refresh stage is eliminated and the data of the forms are submitted to the server asynchronously in order to achieve an interaction process more similar to one of desktop applications. Examples of Form Widgets are: *Button*, *CheckBox*, *RadioButton* and *RichTextArea*.
- *Complex Widgets* are widgets that do not have a counterpart in HTML, but are composed by different HTML components which behavior is managed through JavaScript functions in order to emulate more sophisticated widgets. The developer can also create its own Complex Widgets within the GWT environment and some come with the toolkit, for instance: *Tree* and *MenuBar*.
- *Simple Layout Panels*. Widgets described so far represent the leaves in the hierarchy of the GUI; normally it is not possible to add all the components directly to the root of this hierarchy, but it is recommended to add elements which can, on their part, contain other widgets: these elements are called panels. Simple Layout Panels are the basic panel widgets. Some examples are: *FlowPanel*, *FlexTable*, *Grid* and *DockPanel*.
- *Complex Layout Panels* differ from Simple Layout Panels because they have controls that let the user dynamically change the layout of the contained widgets. Complex Layout Panels are: *StackPanel* and *TabPanel*.
- *Simple Container Panels* are not used, as panels described so far, to mange the layout of the widgets they contain; instead, they provide added functionality as a container of a single child widget. Some Simple Container Panels are: *Composite*, *SimplePanel*, *ScrollPanel*, and *FocusPanel*.
- *Complex Container Panels* can have only one child widget, but they offer also some advanced control behaviors. Examples of Complex Container Panels are: *FormPanel*, *DisclosurePanel*, *PopupPanel*, and *DialogBox*.

For the client-server communication, GWT offers two different solutions: the developer can use the classes belonging to the package (*com.google.gwt.http.client*) to send HTTP request to the server or exploit the *GWT RPC framework* [163].

The mechanism of RPC provided by GWT is based on Java Servlets and involves the generation of client-side and server-side code that allows the effective exchange of serialized Java objects between client and server using the HTTP protocol; the list of Java objects that can be serialized and transmitted with GWT are represented in Table 5.4. When the RPC are used properly, it is possible to obtain an effective separation of responsibilities between client, that manages the graphical tasks, and server, which is involved in the application logic tasks.

Table 5.4: Java objects and primitive types that can be used for client-server communication in GWT.

Type	Description
Java primitive types	`boolean`, `byte`, `char`, `double`, `float`, `int`, `long`, `short`
Wrapper classes	`Boolean`, `Byte`, `Character`, `Double`, `Float`, `Integer`, `Long`, `Short`
Subset Java objects	Only `ArrayList`, `Date`, `HashMap`, `HashSet`, `String`, `Vector`
User defined classes	All classes that implement `isSerializable` or `Serializable` and which attributes are serializable
Array	All the arrays containing serializable classes

With GWT RPC all the calls from the HTML page (i.e. the client) and the server are asynchronous, this means that the client does not idle waiting for the server's response, but continues its execution (in fact the JavaScript engines are typically single-threaded); this leads to a better user experience that can be compared to the one typical of standard desktop applications. The server-side code that is invoked by the client is often called *service*, so the action to perform a RPC is called "invoking a service".

The components of the GWT RPC mechanism are essentially three:

- A remote service (server-side servlet) that runs on the server.
- Client code to invoke that service.
- Java data objects, which will be passed between client and server.

The steps that client and server perform in a typical RPC interaction are the following:

- The user interface calls the service interface on the client-side.
- The client serializes all the objects to be transmitted to the server and calls the remote service (on the server) with an HTTP request.
- The remote service de-serializes the transmitted objects and calls the actual Java implementation of the service (server-side).
- Once the response has been computed the remote service serializes the response objects and all the above steps are executed backwards.

5.2. Literature Mining System

Scientific literature is a primary source of knowledge for biomedical scientists, in fact all recent advances and discoveries are published in the scientific journals and, moreover, published articles contain also information about relevant medical cases observed around the world in the clinical and research practice.

Table 5.5: Number of PubMed indexed articles published from 1950 to 2012.

Year	Published aricles	Year	Published aricles
2012	1013586	1980	278594
2011	997891	1979	280058
2010	929321	1978	270981
2009	867690	1977	260589
2008	828294	1976	253836
2007	779723	1975	247972
2006	742260	1974	234339
2005	696075	1973	230622
2004	635309	1972	227052
2003	590792	1971	223071
2002	560964	1970	218875
2001	543385	1969	214725
2000	529111	1968	207605
1999	490029	1967	191520
1998	470524	1966	179700
1997	452647	1965	176621
1996	453304	1964	161507
1995	443622	1963	141082
1994	432570	1962	125357
1993	421741	1961	119563
1992	413273	1960	111547
1991	407985	1959	109481
1990	406599	1958	108956
1989	399076	1957	111118
1988	382717	1956	106560
1987	364368	1955	108089
1986	346410	1954	105164
1985	332356	1953	108020
1984	315515	1952	107447
1983	306856	1951	102084
1982	293047	1950	82281
1981	281221		

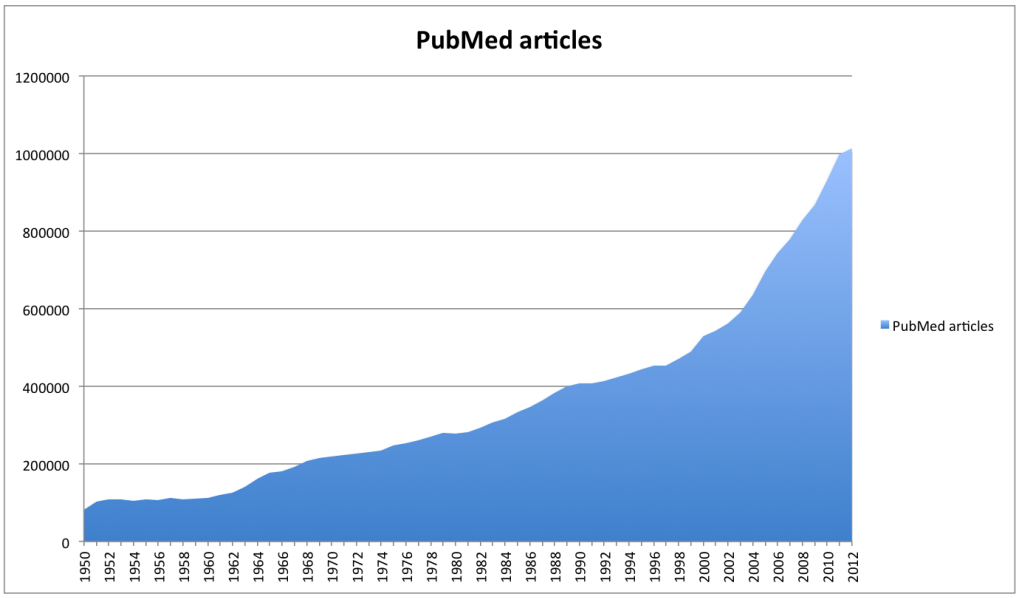

Figure 5.9: Trend of the PubMed publication rate from 1950 to 2012.

The mass of articles that are published every year is impressive; the trend of articles published in PubMed [164] is represented in Table 5.5 and Figure 5.9: in 2012, for the first time ever, this number exceeded the impressive number of 1 million and this trend is not expected to change in the near future. Therefore, for scientists, a knowledge acquisition strategy that involves the manual reading of a large number of articles (also if selected with the many searching tools available – e.g. ENTREZ) cannot be proposed; it is necessary to use tools that can automatically read and understand this huge amount of articles, in order to provide the scientists with summarizing resources that exploit, beyond the available indexing data associated with every publication (e.g. the Mesh terms [63]), also the much more detailed information that comes from the unstructured free-text content of the articles (i.e. title, abstract and full text). One of the most important tasks in the field of automatic analysis of scientific literature is the Information Extraction (IE) task that aims at extracting structured information from unstructured sources. In order to accomplish this work, beyond the development of mining technologies, it is necessary to exploit some kind of structured data source onto which the data extracted from texts are mapped; this structured resources, for the biomedical domain, are typically vocabularies, thesauri, taxonomies and ontologies designed to classify particular types of entities, such as genes, proteins and medical terms, with a variable precision level. The literature mining system described in this thesis has been equipped with tools to extract, from free, unstructured text, concepts that are mapped onto:

- NCBI Gene [165] for genes.
- UniProt – Swiss-Prot [80] for proteins.

- The Human Metabolome Database (HMDB) [166] for metabolites.
- UMLS [152] for biomedical terms including also some of the previous categories (such as genes).

Moreover, a tool to define a set of regular expressions [48] necessary to process, extract patterns and annotate parts of text has been developed.

This chapter presents the several tools that compose the Literature Mining system:

- the Literature Mining Database that stores annotated literature.
- the tools to access PubMed data.
- the GATE text mining plugins to extract information from articles.
- the system that actually performs this IE task.
- the tool that updates the database with the mesh terms of the articles.
- the tool to query the literature database.
- a Literature Based Discovery tool that exploits the Literature Mining system.

5.2.1. The Literature Mining Database

Since the number of potentially interesting articles is high and since the IE process, also when optimized, can be very resource- and time- consuming, we have chosen to develop a persistence layer on which the literature mining system stores its results in order to make them available for future use and that other systems, that need the results of the IE process, can query. This layer has been implemented with a relational database where the basic information about articles and the concepts extracted from their abstracts are stored, along with a standard file system in order to save locally the abstracts as plain text files making them available for further analyses. This persistence layer is called Literature Mining Database (LM-DB).

The LM-DB has been designed and developed with particular attention to the consistency of the contained information and to flexibility, since the data that can be extracted from scientific literature's abstracts are extremely heterogeneous. Moreover, we implemented some facilities in order to easily remove or update the data inside the database; a typical situation where this flexibility is needed happens when the IE tools are updated, because the underlying algorithms are changed or because the structured resources on which they map the extracted data are replaced with new up-to-date versions (the organizations that cure the biomedical vocabularies and taxonomies release new versions at least once a year). In this case the LM-DB provides functionalities to perform a new annotation

of the articles (or a subset of them) limited to one or more types of concepts without losing the data of other types.

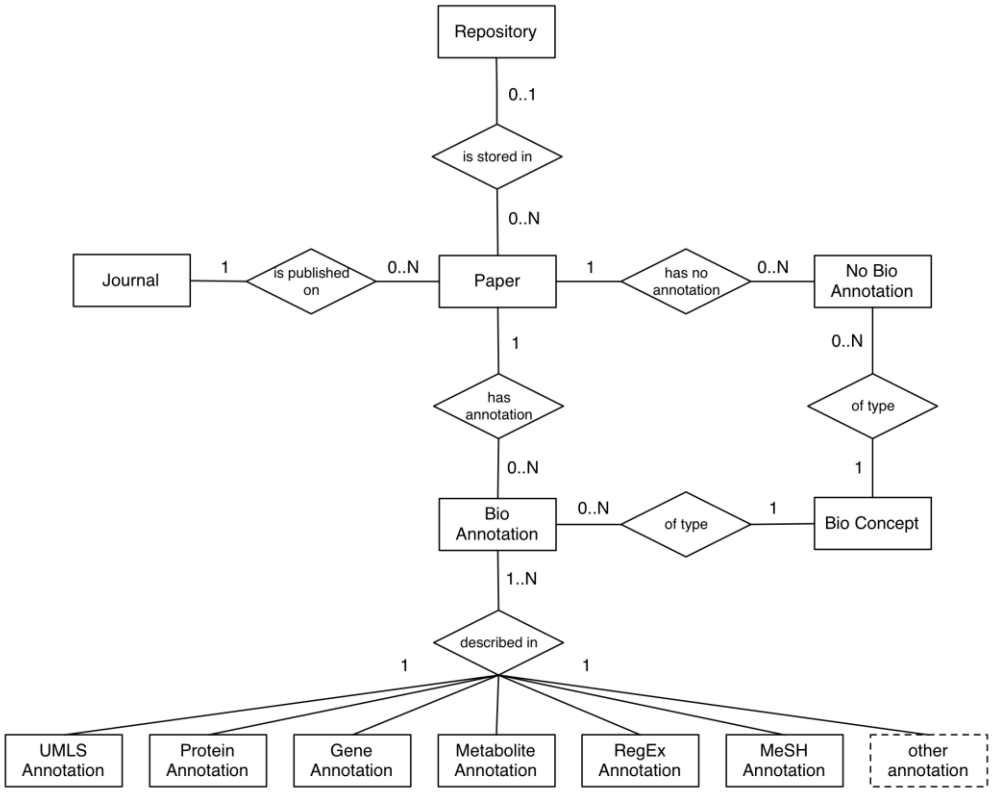

Figure 5.10: The Entity-relationship diagram of the Literature Mining Database.

The overall architecture of the LM-DB is described in Figure 5.10 with an Entity-Relationship diagram. The main table of LM-DB is *Paper*, which attributes are represented in Table 5.6 along with their data types.

The attributes of *Paper* are:

- *pid*, the primary key of the table, that represents the PubMed Unique Identifier of the article;
- *title* that contains the title of the paper;
- *pub_date* with the date of publication of the article;
- *jou_issn* contains the International Standard Serial Number (ISSN) of the journal where the article is published;
- *abs_repository* that is the identification code of the repository (on the file system) where the paper's abstract has been saved;
- *abs_filename* represents the name of the file that contains the abstract of the article;
- *authors* that contains the names of all the authors of the article separated by a comma.

Table 5.6: Attributes and data types of *Paper* table in the LM-DB.

Paper		
Key	**Name**	**Data Type**
K	pid	varchar(10)
	title	varchar(1000)
	pub_date	date
	jou_issn	varchar(10)
	abs_repository	varchar(10)
	abs_filename	varchar(50)
	authors	varchar(2000)

Each record of the *Paper* table is linked, thanks to the *jou_issn* field, to the *Journal* table, which attributes and data types are represented in Table 5.7. Each record of *Journal* has two attributes:

- *j_issn*, the primary key, that represents the ISSN of the journal.
- *name* which contains the abbreviated name of the journal.

Table 5.7: Attributes and data types of *Journal* table in the LM-DB.

Journal		
Key	**Name**	**Data Type**
K	j_issn	varchar(15)
	name	varchar(300)

Obviously, while a single paper is linked to only one journal, the same journal can be linked to more that one paper.

Furthermore, all the papers for which it has been possible to download the abstract (not all the papers in PubMed are associated with an abstract) are linked to the *Repository* table through the attribute *abs_repository*. *Repository*'s attributes and data types are represented in Table 5.8:

- *rep_id*, the primary key, is the unique identification code of the repository within the LM-DB;
- *rep_address* contains the absolute path of the repository inside the file system;
- *comment* is a field that should contain a short description of the papers that are stored in the repository or the purposes for which it has been created.

Table 5.8: Attributes and data types of *Repository* table in the LM-DB.

Repository		
Key	**Name**	**Data Type**
K	rep_id	varchar(15)
	rep_address	varchar(100)
	comment	varchar(2000)

Each paper is therefore connected to no repository, if its abstract is not available from PubMed, or to one repository, if its abstract has been downloaded and stored; each repository is connected (i.e. contains) more than one abstract.

These three tables contain all the basic information about the articles in the LM-DB, while the remaining tables manage the data extracted from the articles. *Bio Concept* is the table that contains the concept types that are managed in the LM-DB; its attributes and data types are shown in Table 5.9:

- *bio_id*, the primary key of this table, contains the unique identification code of the concept type;
- *bio_database* attribute is an optional attribute where the developer can declare which resource stands above the IE process (e.g. "UMLS-2011AA" for the UMLS concepts);
- *description* is another optional field that can contain an arbitrary description of the concept;
- *bio_table* is the name of the specific table in the LM-DB where the concept types are actually stored.

Table 5.9: Attributes and data types of *Bio Concept* table in the LM-DB.

Bio Concept		
Key	**Name**	**Data Type**
K	bio_id	varchar(10)
	bio_database	varchar(20)
	description	varchar(100)
	bio_table	varchar(20)

Bio Annotation is the central table for the annotations in the LM-DB; each annotation performed on the articles, whichever the annotation type, is

represented by a record in this table. The attributes and data types of *Bio Annotation* are represented in Table 5.10:

- *pid*, that is part of the key of the table, represents the PubMed Unique Identifier of the paper which the annotation belongs to;
- *concept_count*, is an integer value that contains the number of times that the annotated concept has been found in the paper;
- *concept_id*, that is part of the key of the table, is the concept identification number in the annotation table specific for its concept type;
- *concept_type*, also part of the key, contains the unique identification code for the concept type;
- *concept_location*, that is also part of the key, contains the part of the paper from where the annotation has been made (e.g. "abstract", "full-text", "title" or, in the case of MeSH annotations, "MEDLINE description").

Table 5.10: Attributes and data types of *Bio Annotation* table in the LM-DB.

Bio Annotation		
Key	**Name**	**Data Type**
K	pid	varchar(10)
	concept_count	int
K	concept_id	int
K	concept_type	varchar(10)
K	concept_location	varchar(10)

Each record of *Bio Annotation* is connected to one paper (in fact *pid* is part of the key) and a paper can be linked to more records in *Bio Annotation*, one for each its unique annotation (duplicate annotations are managed with the *concept_count* field). Moreover each annotation has one single *concept_type* (from *Bio Concept*) while, obviously, a concept type can characterize more than one annotation. Given the concept type, it is possible (thanks to the *bio_table* attribute of *Bio Concept* table) to link the generic annotation to the specific table where all its attributes are specified; therefore, each annotation from *Bio Annotation* is linked to one record belonging to a specific annotation table. It is important to note that, when two or more generic annotations refer to the same specific annotation (exactly the same, for all the attributes describing it), the system, that avoids duplications, associates the same record of the specific annotation table to all the generic annotations in *Bio Annotations* that refer to it.

The specific annotation tables (the ones at the bottom of Figure 5.10) are extremely heterogeneous because the attributes that characterize them depend on the very nature of the annotation itself. The only attribute that unites all the specific annotation tables is their primary key: *concept_id*, that is the unique identification number of the specific annotation within its table. By way of example in Table 5.11 are represented the attributes and the concept types of the annotation table specific for UMLS concepts: *UMLS Annotation*. Beyond the primary key, the other attributes are:

- *cui*, the Concept Unique Identifier of the UMLS concept;
- *umls_name*, the official name of the UMLS concept;
- *semantic_type*, the semantic type of the concept;
- *source_vocabulary*, the vocabulary from which the concept comes;
- *tui*, the Type Unique Identifier of the semantic type;
- *string*, the part of the text annotated with this annotation.

Table 5.11: Attributes and data types of *UMLS Annotation* table in the LM-DB.

UMLS Annotation		
Key	**Name**	**Data Type**
K	concept_id	int
	cui	varchar(10)
	umls_name	varchar(2000)
	semantic_type	varchar(100)
	source_vocabulary	varchar(15)
	tui	varchar(5)
	string	varchar(2000)

The last table that composes the LM-DB is *No Bio Annotation*; exactly as for *Bio Annotation*, each record of this table is connected to a single paper. The attributes and related data types of *No Bio Annotation* are represented in Table 5.12; all the three attributes are key for the table:

- *pid* represents the PubMed Unique Identifier of the paper;
- *concept_type* contains the unique identification code for the concept type;
- *concept_location* contains the part of the paper which the record is referred to.

Table 5.12: Attributes and data types of *No Bio Annotation* table in the LM-DB.

No Bio Annotation		
Key	**Name**	**Data Type**
K	pid	varchar(10)
K	concept_type	varchar(10)
K	concept_location	varchar(10)

A single record of *No Bio Annotation* table means that for the paper identified by its *pid* in the specific *concept_location* (e.g. in the abstract), no annotation of the specific *concept_type* has been found. In other words, the system has analyzed the paper looking for the specific type of concepts, but did not find anything. If the strategy of tracking also IE processes that did not carry to any result was not performed, in the LM-DB it would not be possible to discriminate from papers that have never been analyzed for the specific concept type from those that do not contain any concept of that type.

In the following we will make an example of how the LM-DB is populated starting from a paper that has been analyzed in order to extract the contained UMLS concepts and the gene names. This example paper is not a real PubMed article but, for shortness sake, it is composed by a single phrase taken from the incipit of the Wikipedia "Dilated cardiomyopathy" page. This example paper is assigned with PubMed identifier "*EX123456*", authors "*Matteo Gabetta*" and "*John Doe*", publication date "*December 25th 2011*", journal ISSN "*0000-000X*", journal title "*Average Journal*" and its abstract is:

> *<<Dilated cardiomyopathy or DCM is a condition in which the heart becomes weakened and enlarged and cannot pump blood efficiently. The decreased heart function can affect the lungs, liver, and other body systems.>>*

Figure 5.11 represents the Gate GUI with the document analyzed to extract UMLS concepts (in red) and protein names (not present). Now we will show how this document along with its annotations is stored in the LM-DB:

1. when the paper is downloaded from PubMed, a record in *Paper* is created, see Table 5.13.

Table 5.13: Sample record of *Paper* table.

pid	title	pub_date	jou_issn
EX123456	Dialted cardiomyopathy	25/12/11	0000-000X

abs_repository	abs_filename	authors
R001	EX123456.txt	Gabetta Matteo, Doe John

2. if the journal with ISSN "0000-000X" was not present in the *Journal* table, the record in Table 5.14 is added.

Table 5.14: Sample record of *Journal* table.

j_issn	name
0000-000X	Average Journal

3. Since the paper was downloaded from PubMed with its abstract, this has been stored in the file system. In particular, the abstract has been saved in the repository "*ROO1*", thus, the record shown in Table 5.15 is present in *Repository*.

Table 5.15: Sample record of *Repository* table.

rep_id	rep_address	comment
R001	~/repositories/REP_001/	this is an example

4. In this example we deal only with UMLS concepts and protein, therefore *Bio Concept* has only two records, see Table 5.16.

Table 5.16: Sample records of *Bio Concepts* table.

bio_id	bio_database
UMLS	UMLS2011AA
PROT	Uniprot 2010

description	bio_table
Unified Medical Language System	UMLS Annotation
UniProt - Swiss-Prot	Protein Annotation

5. From FIGURE 6.10 you can recognize eight UMLS annotations on the paper's abstract; *Bio Annotation* is represented in Table 5.17.

Table 5.17: Sample records of *Bio Annotation* table.

pid	concept_count	concept_id	concept_type	concept_location
EX123456	1	100	_UMLS_	abstract
EX123456	1	101	_UMLS_	abstract
EX123456	1	102	_UMLS_	abstract
EX123456	1	103	_UMLS_	abstract
EX123456	1	104	_UMLS_	abstract
EX123456	1	105	_UMLS_	abstract
EX123456	1	106	_UMLS_	abstract
EX123456	1	107	_UMLS_	abstract

6. The specific annotation table *UMLS Annotation* will have a record for each "*concept_id*" in *Bio Annotation* with the value of "*concept_type*" set, in this case, to "*_UMLS_*" (see Table 5.18).

Table 5.18: Sample records of *UMLS Annotation* table.

concept_id	cui	semantic_type
100	C0007193	Disease or Syndrome
101	C1281570	Body Part, Organ, or Organ Component
102	C1293134	Therapeutic or Preventive Procedure
103	C0229664	Body Substance
104	C0232164	Organ or Tissue Function
105	C0024109	Body Part, Organ, or Organ Component
106	C1278929	Body Part, Organ, or Organ Component
107	C0392912	Body System

umls_name	source_vocabulary	tui	string
Cardiomyopathy, Dilated	MTH	T047	Dilated cardiomyopathy
Entire heart	MTH	T023	heart
Enlargement procedure	MTH	T061	enlarged
peripheral blood	MTH	T031	blood
Cardiac function	SNOMEDCT	T042	heart function
Lung	MTH	T023	lungs
Entire liver	MTH	T023	liver

Entire body system	MTH	T022	body system

7. The other concepts that the system tried to extract from the article are proteins, but no protein name was found within the abstract. Thus in *No Bio Annotation* there will be a single record represented in Table 5.19 meaning that no proteins have been found in the article's abstract.

Table 5.19: Sample record of *No Bio Annotation* table.

pid	concept_type	concept_location
EX123456	_PROT_	abstract

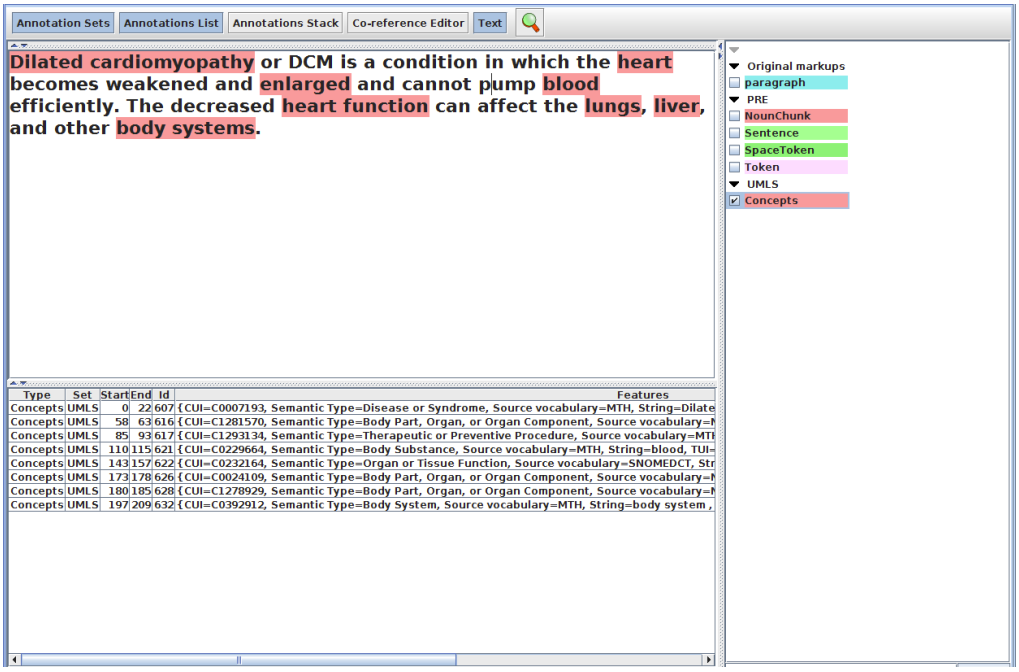

Figure 5.11: The example document with the extracted UMLS concepts in the Gate GUI.

5.2.2. Literature Acquisition

The first step in the global process of analyzing scientific literature and exploit the achieved results, is the acquisition of the papers' data from online repositories. In the developed system the primary source of literature is PubMed that we accessed with the ENTREZ Utilities and, specifically, with SOAP based web services. As all the implemented software of the present work, also the literature acquisition system has been developed with Java programming language and, therefore, we used the E-Utilities

Java APIs to access PubMed. The overall architecture of the Literature Acquisition System is represented in Figure 5.12.

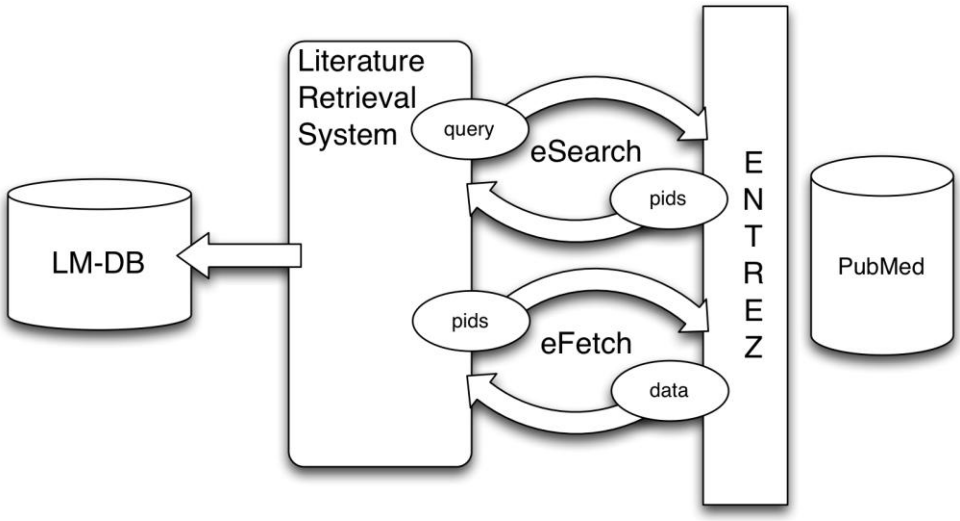

Figure 5.12: The overall architecture of the Literature Mining system.

For the sake of the present work, we focused mainly in the analysis of papers' abstracts; in fact although full-texts may be a better source of biologically relevant data, abstracts have proven to provide the best ratio of keywords per total of words [167]. Furthermore, full texts are not always directly available from PubMed, sometimes they are not free of charge and their analysis implies many issues, such as the correct parsing of different PDF documents' formats, that go beyond the scopes of the present work. Alongside with the abstracts, another data source related to the papers is constituted by the MEDLINE indexing data associated with each article and, in particular, by its MeSH terms.

The literature acquisition part of the system is mainly composed by two Java classes:

- *PubmedSearch* that defines the query to search PubMed;
- *PubmedFinder* that actually performs the query and stores the results in the LM-DB.

PubmedSearch is a very simple object with three attributes:

- *query* represents the object of the query to be performed on PubMed;
- *startDate* is the start of the publication interval of the articles;
- *endDate* is the end of the publication interval of the articles.

The object of the query is always defined when the *PubmedSearch* object is instanciated while the dates, if not provided, are set respectively to an arbitrary low date for *startDate* (1900/01/01) and an arbitrary high date for *endDate* (3000/12/31), in order to practically loosen their constraints.

The core of the literature acquisition is in *PubmedFinder*. This class, when instanced, is provided with:

- a *PubmedSearch* object to define the query object and dates;
- *articlesNum*, the maximum number of papers to acquire;
- *repository_id*, the identification code (in the LM-DB) of the repository where the papers' abstract have to be saved;
- *config*, that is the reference to an XML file that contains the coordinates and access parameters to the LM-DB.

The typical workflow of the literature acquisition process implies the use of two ENTREZ-Utilities: *eSearch* and *eFetch*. After the *PubmedFinder* object has been instanciated and the query defined, the system uses the E-Utilities APIs to run the *eSearch* utility on PubMed and exploits the History server in order to save the result of *eSearch* for the next phase.

Now, thanks to the use of the *WebEnv* and *query_key* parameters, it is possible to set up an *eFetch* query without the need of passing, as input, the PubMed ids produced by *eSearch*. For each article identified the system downloads the related data:

- The title of the article.
- The abstract of the article.
- The creation date of the article.
- The ISSN code of the journal where the article is published.
- The name of the journal.
- The list of the authors' names.

None of this information is strictly necessary in order to add the article to the LM-DB; in fact, it can happen to retrieve articles without the information on their publication journal or about the authors, but, once the system knows their Unique Identification Number, it is possible to insert them in the LM-DB. The most typical case of articles with partial information is constituted by those without an abstract downloadable from PubMed (this is typical for many old publications). In this case the system creates the corresponding record in the *Paper* table in the LM-DB, but leaves empty the fields that regard the abstract location in the file system ("*abs_repository*" and "*abs_filename*"). This is a good solution to keep trace of all the considered articles and, moreover, to exploit also those data that characterize the publication and that are not derived from the abstract, such as the MeSH terms.

The downloaded data are thus exploited in order to insert the article in the LM-DB and, in particular, the literature acquisition part of the system writes the tables of LM-DB that contain the basic information about the

paper: *Paper* and *Journal*. When the literature has been acquired it is possible to perform its analysis in order to extract the contained concepts; it is important to note that the remaining parts of the systems work directly on the LM-DB, therefore, if a subset of literature has not been stored previously, it is impossible for them to work on it; anyway populating the LM-DB can be done progressively and the fact of adding new articles in a second time does not constitute an issue.

5.2.3. Text Mining

With the LM-DB populated with adequate set of articles (ideally the whole corpus of PubMed publications) it is possible to perform the central task of the literature mining system: the IE process that aims at extracting structured knowledge from the unstructured source constituted by the abstracts of the retrieved articles. This process, represented in Figure 5.13, is cyclic, in fact the LM-DB provides the inputs for the IE tools (i.e. the articles' data) and receives its outputs (the extracted concepts) to store them. Also these IE tools have been implemented with the Java programming language, exploiting the GATE APIs (Section 5.1.2). This implementation strategy allowed to make the mining process completely automatic, without the need of using GATE's Graphical User Interface that has been employed in the development phase and for debugging purposes.

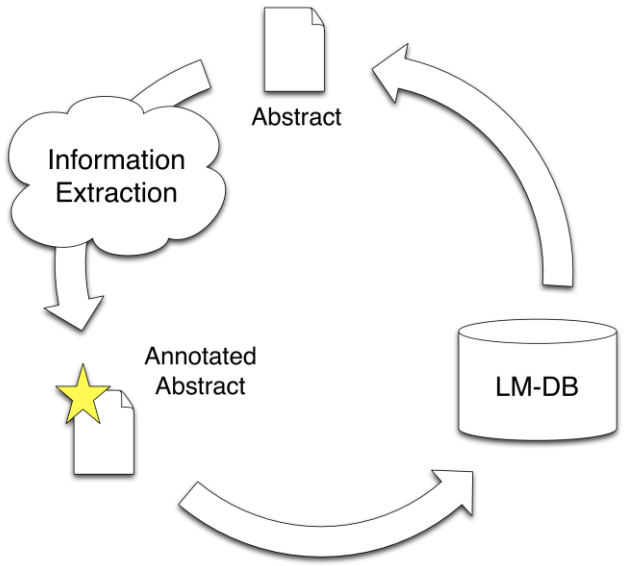

Figure 5.13: The cyclic process of Text Mining in the Literature Mining system.

The concept types that the system manage to extract from literature are:

- Gene names
- Protein names
- Metabolite names
- UMLS concepts
- MeSH terms
- User defined regular expressions.

For each concept type a specific GATE Processing Resource (called plugin) has been developed that manages its extraction; moreover, an additional plugin that manages the acronyms inside the text and that is exploited by some of the concepts extraction plugins has been implemented. These Processing Resources are:

- the *Acronym Finder*, that identifies acronyms in the text in order to better disambiguate the extraction of gene and protein names;
- the *Gene Finder*, that extracts gene names exploiting NCBI Gene;
- the *Protein Finder*, that extracts protein names exploiting UniProt – Swiss-Prot;
- the *Metabolite Finder*, that extractss metabolite names exploiting HMDB;
- the *UMLS Finder*, that extract UMLS concepts from text;
- the *RegEx Finder*, that, provided with a set of regular expressions and rules to annotate them properly, extracts them from the text.

The extraction of MeSH terms from literature does not need a dedicated text mining plugin because MeSH terms are already a structured data source and the only thing the system needs to do is their acquisition from PubMed (again with ENTREZ Utilities) and the proper uploading into the LM-DB.

5.2.3.1. The Acronym Finder

The first plugin that we will describe is the *Acronym Finder*, the processing resource aimed at identifying typical *complete name/acronym* patterns. Examples of these patterns, which do not force the complete name to be immediately adjacent to its acronym, are:

Dilated Cardiomyopathy (DCM)
Dilated cardiomyopathy, abbreviated as (DCM)
Medical subject headings (MESH)
Medical Subject Headings (MeSH)

The results of the acronym identification process is typically used by some IE plugins in order to understand if an acronym represents the shortened

form of a molecule or not. In fact, one of the most challenging tasks in the extraction of biological entities from texts is the fact that frequently the shortened name of a gene or a protein is also associated with plenty of other abbreviations that have nothing to do with that biological entity; for example "Desmin" is a gene which is often abbreviated as "DES", but this acronym is also used for a pharmacological substance called "Diethylstilbestrol"; the goal that the *Acronym Finder* is asked to reach is to make possible to the IE plugins to recognize, when they have to evaluate the word "DES", if it is used as the shortened gene name or as the pharmacological substance, by providing a common annotation that keeps track both of the short and the extended form. Therefore, when the IE plugin (in this case the *Gene Finder*) faces the word "DES", it checks if this word has been recognized as an acronym (i.e. has been identified within a typical *complete name/acronym* pattern), evaluates its extended form and, if they are both referred to the same entity (i.e. a gene) they are both annotated, otherwise they are both discarded.

The steps the *Acronym Finder* performs to identify *complete name/acronym* patterns in the text are three:

1. it detects all the parts of text between parenthesis.
2. some rules are applied in order to filter only those character sequences that are suitable to be an acronym. These empirical rules are:
3. the sequence's length must be at least 2
4. the maximum length of the character sequence is 10
5. the sequence must have at least one uppercase letter
6. the sequence must not contain any of this characters: "\′=<>:;?!°"
7. the character sequences that have passed the second step are cleaned of some characters that may be part of the acronym, but that do not affect its structure; these characters are: "-_." and whitespace.
8. the final step consists in the progressive comparison of the acronym with the tokens that precede it, looking for a word sequence suitable to be represented by the acronym. If such a word sequence is identified the text is annotated with a new annotation that brings the information of the acronym and its extended name.

5.2.3.2. The Gene Finder

The *Gene Finder* is the plugin aimed at extracting gene names from free text. This plugin relies on the NCBI Gene database that counts more than 11.000.000 genes [168], each one assigned with a unique identifier number, its official name, the species identification number, and eventually with a set of unofficial short and long names. This plugin works on texts that have

been previously tokenized and adds to the GATE document annotations on eventually contained gene names along with the corresponding identification number and official name.

The *Gene Finder* takes several parameters as input in order to tune the extraction process on the specific needs of the user. These parameters are:

- a list of user-defined stop-words: common English words which correspond to gene names (typically, but not always unofficial names) that the system is asked to ignore;
- a boolean flag to establish if the *Gene Finder* has to check acronyms in the way that is described for the *Acronym Finder;*
- a boolean flag to establish if the plugin has to consider only those genes that are associated with a specific species (e.g. "9606" for homo sapiens);
- eventually the considered species identification number;
- a boolean flag that establishes if the *Gene Finder* considers exclusively official short and long names, or if also unofficial names have to be taken into account;
- a boolean flag which, if set *true*, limits the analysis only to those names (typically short names) which are capitalized;
- a boolean flag to establish if the comparison between names in text and gene names in the database is case-sensitive or less;
- the minimum length for a gene name, to avoid very short character sequences to be identified as genes.

The IE process follows these steps:

1. each token is considered and, first, the *Gene Finder* queries the database (limiting to the gene names that are coherent with the setting parameters of the plugin) to know if there is at least one gene name that starts with the token in exam (this operation is called "soft query" because it searches for partial matches and not exact matches between the token in exam and the database).
2. if the token does not match the soft query, then the token and any character sequence starting with it cannot be gene names. The token is discarded and the plugin goes back to the previous step with the next token.
3. if the token matches the soft query, the plugin progressively adds to it the following tokens repeating each time the soft query and the aggregated word. The next token is added only until the aggregated word matches the soft query.
4. once the aggregated word fails the soft query, the *Gene Finder* considers the last aggregated word that matched the soft query and performs a new query on the database, this time in order finding an exact match between the aggregated word and a gene name (this query is called *exact query*).

5. if the considered characters sequence fails the exact query the plugin goes back to point (1) with the next token.
6. if the considered character sequence matches the exact query then the plugin has identified a possible gene name in the text.

Once all the tokens have been considered, the *Gene Finder* has a list of possible gene names extracted from the text, but, in order to achieve the right results, it has to perform some further steps to refine this result set:

7. overlapping annotations are deleted in favor of the longest one that is maintained.
8. the stop-words list is used to delete annotations that match with the contained words.
9. identified gene names that are also roman numbers are deleted from the set of possible annotations
10. identified gene names that are completely numeric are not considered.
11. finally, if the Gene Finder has to check the acronyms, gene names in the text that are also part of a complete name/acronym couple are evaluated; only if the complete name and the acronym have been recognized as gene names and only if they have been annotated as the same gene, the plugin maintains them in the set of possible gene names extracted.

Finally the GATE document is added with the remaining annotations; each gene annotation is characterized by:

- the unique identifier number of the gene.
- the official name of the gene.
- the name identified in the text as the gene.

5.2.3.3. The Metabolite Finder

The *Metabolite Finder* is the plugin aimed at extracting metabolite names from free text. This plugin relies on the Human Metabolome Database (HMDB) that counts more than 40.000 entries [169], each one assigned with a unique identifier code, its official name and eventually with a set of synonyms. This plugin works on texts that have been previously tokenized and adds to the GATE document annotations on eventually contained metabolite names along with the corresponding identification code and official name. To set up the *Metabolite Finder* some parameters have to be specified:

- a list of user-defined stop-words and
- a boolean flag to establish if the *Metabolite Finder* exploits the acronyms identified by the *Acronym Finder*.

The *Metabolite Finder* is directly derived from the *Gene Finder*, therefore the steps that are performed to identify metabolite names within the text are the same, except for the fact that this plugin relies on a different database and, therefore, the sql-queries performed are customized for the specific task. In particular all the steps to obtain the list of possible metabolite annotations are the same (i.e. steps 1-6 of the *Gene Finder*). Moreover the *Metabolite Finder* performs also step 7 (to manage overlapping annotations), 8 (to delete metabolite names that belong to the stop-words list) and 11 (to exploit the information coming from the analysis performed by the *Acronym Finder*).

Finally the plugin writes the remaining metabolite annotations in the GATE document, each one characterized by:

- the unique identifier code of the metabolite,
- the official name and
- the name identified in the text as the metabolite.

5.2.3.4. The Protein Finder

The *Protein Finder* is the plugin that extracts protein names from free text. This plugin relies on the Uniprot – Swiss-Prot database that counts more than 75.000 entries [170], each one assigned with a unique identifier code (called *accession*), its recommended name and eventually with a set of synonyms that can be long or short names. This plugin works on texts that have been previously tokenized and adds to the GATE document annotations on eventually contained protein names along with the corresponding accession and recommended name. To set up the *Protein Finder* some parameters have to be specified:

- a list of user-defined stop-words,
- a boolean flag to establish if the *Protein Finder* exploits the acronyms identified by the *Acronym Finder,*
- a boolean flag to establish if the plugin has to consider only those proteins that are associated with a specific species,
- eventually the considered species identification number.

Also the *Protein Finder* is derived from the *Gene Finder* and the first steps to identify protein names in the text (steps 1-6) are the same; obviously the actual queries performed on the database are customized in order to work properly on the protein database. Since the *Protein Finder* shares also some parameters with the *Gene Finder*, it performs some of its final steps to clean the set of possible annotations: step 7 (overlapping annotations), 8 (stop-words) and 11 (acronym management).

The final annotations written on the GATE document contain the following features:

- accession number of the protein,
- recommended name of the protein,
- name found within the text,
- a boolean flag that is "true" if the name found is the recommended name,
- a qualifier for the name found ("short" or "long").

5.2.3.5. The UMLS Finder

The *UMLS Finder* is a very general IE plugin that exploits the UMLS Metathesaurus in order to extract from text those concepts that are represented in the Metathesaurus itself. Considered the dimension of UMLS and its general purpose nature, it is important to focus the IE task on specific source vocabularies in order to obtain reliable results; in fact the Metathesaurus is composed by about 2.900.000 concepts with more than 11.200.000 names associated [171].

This plugin uses some of the annotations done by other processing resources:

- token annotations performed by the *ANNIE English Tokenizer* improved with the *GATE Morphological Analyzer* (that adds the information on the lemma of the word),
- noun phrase annotations done with the *Noun Phrase Chunker*.

The *UMLS Finder* needs to be set up with some input parameters in order to work properly:

- a list of valid Semantic Types to which the extracted concepts have to belong;
- a list of source vocabularies of the Metathesaurus onto which the research process is performed;
- a list of Concept Identification Numbers (CUIs) (called stop-cui list) that are not to considered in the IE process.

The extraction of UMLS concepts follows these steps:

1. the plugin considers one chunk (i.e. noun phrase) per time;
2. within each chunk, considering the single tokens, the *UMLS Finder* creates all the possible sub-chunks;
3. each sub-chunk is used, as the concept name, to build the query that is performed on the Metathesaurus; moreover the query is formed also by constraints on the source vocabularies to which the retrieved concepts have to belong to;
4. if the query matches a concept, its Semantic Types are retrieved and, if none of them belongs to the list of valid Semantic Types,

the *UMLS Finder* discards it and returns to step (3) with the next sub-chunk;

5. the plugin checks if the CUI of the identified concept belongs to the stop-cui list. In that case the concept is discarded and the execution goes back to step (3) with the next sub-chunk;
6. if at least one of the concept's Semantic Types belongs to the list and the concept's CUI does not belong to the stop-cui list, the plugin queries the Metathesaurus to complete the information that characterizes the UMLS annotation. The data retrieved in this process are:
   - the name of the source dictionary from where the concept comes; in the frequent case of more than one dictionary including the concept, the *UMLS Finder* selects the one with the best ranking in the Metathesaurus;
   - the preferred name that the concept has in the best ranked dictionary of origin.
7. the plugin adds the new annotation to a list of possible annotations that, after all these steps have been completed, is post-processed in order to determine which annotations will be actually added to the GATE document;
8. if the query at step (3) does not match any results, the *UMLS Finder* exploits the results of the *GATE Morphological Analyzer* and converts all the plural nouns, that constitute the sub-chunk in exam, into their lemma (i.e. their singular form). This operation aims at standardizing the sub-chunk because singular forms are commonly more used than plural forms in the biomedical data sources. Now if the query with the "transformed" sub-chunk matches, the execution passes through the same steps (steps 4-7).

The *UMLS Finder* has now to refine the list of possible UMLS annotations in order to delete the overlapping ones giving place to the more extended ones. The remaining annotations are then added to the GATE document with the following features:

- Concept Unique Identifier of the concept
- Semantic Type of the concept
- Type Unique Identifier of the concept
- best ranked source vocabulary
- preferred name in the best ranked vocabulary
- string identified as the concept in the text

5.2.3.6. The RegEx Finder

The last processing resource is the *RegEx Finder*; a plugin provided by the user with a list of Regular Expressions (RegEx) [48] that it tries to identify within the text. Each RegEx is associated with an annotation name assigned to the part of the text that matches the RegEx and a series of feature/value pairs that characterize the annotation. The RegEx, annotation name and feature/value pairs constitute a rule for the *RegEx Finder*. A simple example rule that can be passed to the plugin is the following:

```
(?i)mg Measure     type=mass   name=milligram
```

The first element is the RegEx, it is matched by the string "*mg*" without considering the case of its letters, then also the string "MG" will match the RegEx. The second element (*Measure*) is the name of the annotation that will be done to the parts of the text that match the RegEx. After the annotation's name, the rule is composed by an arbitrary number of feature/value pairs that characterize the annotations on the parts of the document that match the RegEx.

The *Regex Finder* is useful when it is necessary to extract from text a limited set of concepts that can appear in different forms (in fact the RegEx allow a good flexibility) and when there is not a structured data source onto which map the extracted concepts, or when it would not be convenient to develop a dedicated processing resource.

5.2.4. Literature Analysis

With the LM-DB populated with the literature of interest, ideally all the articles retrievable from PubMed, it is now possible to use the IE plugins along with the whole GATE framework in order to extract from the abstracts the different types of concepts and store them in the LM-DB. For the concept extraction task we have developed a single pipeline for each type of concept to extract, exploiting the Java GATE APIs. This strategy has been followed in order to easily analyze the single types of concepts (e.g. genes, proteins, UMLS concepts) autonomously. Anyway, all the described IE plugins, with the exception of *RegEx Finder* which requires as input a plain text, work on texts already annotated by one or more standard text mining plugins (such as a tokenizer, POS-tagger and chunker). All the IE pipelines are depicted in Figure 5.14, where, instead of five separated pipelines, it is represented a single pipeline which splits its execution into five branches, after the shared processing resources.

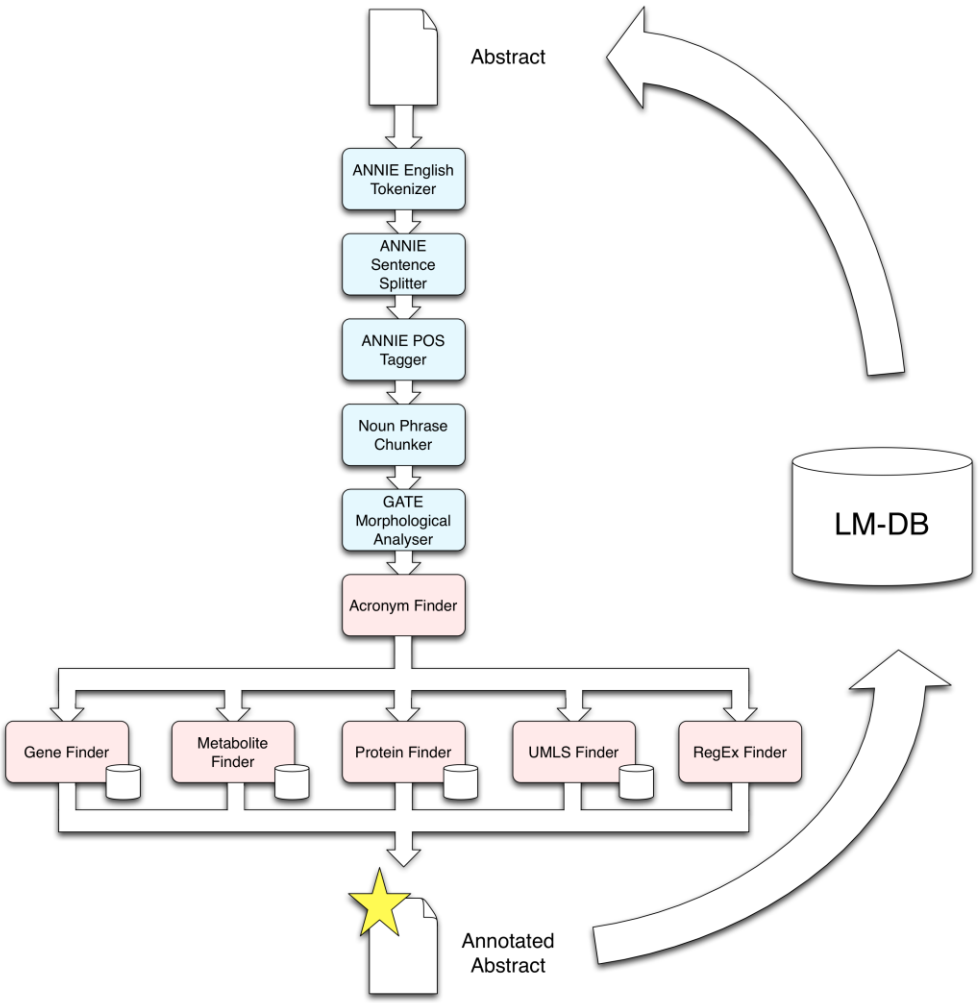

Figure 5.14: The comprehensive Information Extraction pipeline of the Literature Mining system.

In the following the single plugins are briefly recalled:

- The *ANNIE English Tokenizer* identifies tokens in the text.
- The *ANNIE Sentence Splitter* annotates single sentences.
- The *ANNIE POS Tagger* adds to each token its grammatical class (Part Of Speech).
- The *Noun Phrase Chunker* identifies noun phrases in the text.
- The *GATE Morphological Analyzer* assigns to each token its lemma.
- The *Acronym Finder* identifies typical *complete name/acronym* patterns.
- The *Gene Finder* extracts gene names exploiting NCBI Gene.

- The *Metabolite Finder* extracts metabolite names exploiting HMDB.
- The *Protein Finder* extract protein names exploiting Swiss-Prot.
- The *UMLS Finder* extracts UMLS concepts.
- The *RegEx Finder* matches the text with user-defined regular expressions.

The execution of each IE pipeline accomplishes several steps in order to acquire an abstract from the LM-DB and write back the extracted concepts. Each IE process needs some parameters to work properly:

- the plugins input parameters of the GATE pipeline.
- the coordinates and access parameters of the LM-DB.
- the name of the concept to extract (the *bio_id* attribute of *Bio Concept* table in LM-DB). Through this information the IE system can query LM-DB to know the specific annotation table of destination for the extracted concepts (*bio_table* attribute of *Bio Concepts*).
- a boolean flag, called *overwrite*, that determines the behavior of the IE systems with papers already analyzed for the specific concept.
- the list of PubMed unique identifiers (pids) of the articles to analyze.

The first step performed, after the initialization of the IE system with the input parameters, is the loading process of the plugins involved in the specific analysis; each plugin is initialized with the required input parameters that the IE system extracts from a configuration XML file. Once all the plugins have been loaded, the system composes the IE pipeline positioning them in the right sequence, so that the output (i.e. a partially annotated GATE document) of one plugin is the input for the following and so on. To avoid the system to be blocked for a long time (the IE process, depending on the involved plugins, can be a very time-consuming task) the execution is divided in many "sub-executions" each one for at least 100 abstracts. Now the system evaluates the *overwrite* parameter: if it is "false", all the pids of articles already analyzed for the specific concept are removed from the list of pids.

Then, querying the LM-DB tables *Paper* and *Repository*, the IE system is able to assign to each article its actual address in the file system and to retrieve the plain text file containing its abstract. Once all the abstracts are available, the system creates a GATE corpus that is analyzed, one article in turn, by the text mining pipeline.

When all the abstracts are annotated, the system is ready to write back the extracted concepts to the LM-DB. Before this operation it is necessary to evaluate again the *overwrite* flag: in fact, if it is "true", the system deletes all the records of the analyzed articles for the specific concept from the tables *Bio Annotation* and *No Bio Annotation*, so that, relatively to the concept of interest, they come back to the not-analyzed state.

The update process of the LM-DB with the new annotations of the analyzed articles follows these steps:

- the system checks if, in the specific annotation table (e.g. *UMLS Annotation* for the concepts extracted by the *UMLS Finder*), another identical annotation record exists (typically the same annotation performed on a previously analyzed article); in this case the system avoids the duplication of the record and simply adds, in the generic annotation table *Bio Annotation*, a new tuple that refers to the new article and the old specific annotation.
- in the case of an annotation that is not yet in the specific annotation table, the system simply adds a new record to this table and to the generic annotation table *Bio Annotations*.
- if no concept is extracted, the system updates, instead of the generic and specific annotation tables, only the table *No Bio Annotation* for the specific concept and article's abstract.

5.2.4.1. The Mesh Finder

The only part of the IE system that differs from the standard implementations described in the previous Section is the MeSH extraction system. In fact, MeSH terms are structured data and no text mining process is needed in order to associate an article with its MeSH terms. The acquisition process of these data is very similar to the literature acquisition process, since the primary source of MeSH terms is PubMed that is accessed with the ENTREZ Utilities. The typical workflow of the Mesh Finder is shown in Figure 5.15.

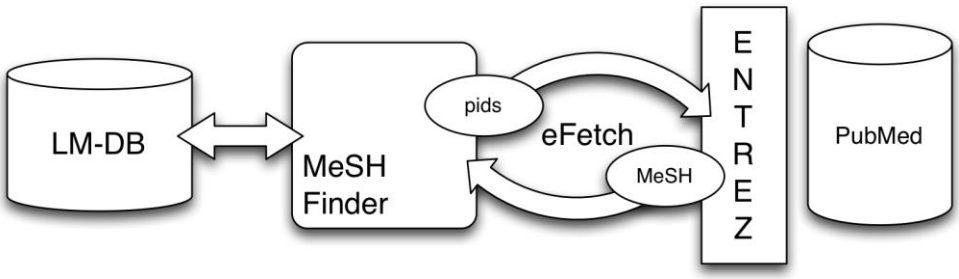

Figure 5.15: The workflow of the Mesh Finder

The input parameters to set up the MeSH retrieval process are:

- the list of PubMed unique identifiers of the articles which MeSH terms have to be retrieved;

- the coordinates and access parameters of the LM-DB;
- the *bio_id* attribute of *Bio Concept* table in LM-DB relative to MeSH concepts. With this information the IE system can query LM-DB to know the specific annotation table for MeSH terms;
- the *overwrite* flag that works in the same way as described for the other parts of the IE system.

The MeSH retrieval process exploits the *eFetch* tool; after the initialization with the set up parameters, the system evaluates the *overwrite* flag: if it is set "false" those pids that match to articles whose MeSH terms have already been added in the LM-DB are discarded for further steps. The *eFetch* service is called, but, differently from the literature retrieval system, the data downloaded from ENTREZ are only the MeSH terms relative to the articles queried. Once all the MeSH terms have been downloaded, the system, if the *overwrite* parameter is "true", deletes from LM-DB all the previously retrieved MeSH concepts. Finally, the system is able to add to the LM-DB the MeSH concepts downloaded, paying attention to avoid duplications in the specific annotation table (*Mesh Annotation*) as described for the other arts of the IE system.

5.2.5. Interrogation Tools

Along with the LM-DB, it has been developed also a utility tool to interrogate the data stored inside the system. This tool, implemented in Java programming language, takes in input:

- the list of PubMed ids of a set of selected articles;
- the coordinates and access parameters of the LM-DB.

The interrogation tool then queries the LM-DB for the selected articles and returns as output:

- the number of requested articles that are present in the LM-DB;
- the percentage of present articles on the total amount of requested articles;
- for every type of concept present in the LM-DB (achieved from the *Bio Concept* table):
    - the number of requested articles that have been analyzed for such a concept type;
    - the number of requested articles that have associated at least one concept of the specific type;
    - a table containing all the concepts found in the literature set of interest with all the attributes that characterize the concept type (the attributes of the specific annotation table of each concept type) and, for each concept, the list of articles that have been associated the concept.

The tables relative to the different concept types are in Comma Separated Values (CSV) format so that they can be imported into many applications (such as Microsoft Excel) or can be parsed directly by some programming language to exploit their content.

5.2.6. Literature Based Discovery

The Literature Based Discovery (LBD) tool that we have developed within the Literature Mining system has been entirely built on top of Java-based technologies integrated with freely accessible datasets and web services. The concepts, core of data representation inside the LBD system, are codified using UMLS, their persistence is entrusted to the LM-DB, the literature access is performed with the *PubmedFinder* and finally the Graphical User Interface of our LBD system has been developed with Google Web Toolkit in order to easily expose the Java-based system on the Web.

The overall workflow of the discovery process underlying the LBD system is represented in Figure 5.16; the approach is based on the open discovery paradigm that, starting from a source concept, tries to discover related knowledge (i.e. other concepts) that have never been directly associated with the starting one (i.e. co-cited inside an abstract), but prove to have a strong relationship with it in terms of intermediate concepts directly associated with both.

The first step the user is asked to perform, is to choose the starting concept to query (*A concept*); in practice the user provides the name of the concept and the system tries to match it in the UMLS; if the matching succeeds, the system asks the user to choose, if necessary, one or more synonyms of the A concept that will be included in the PubMed query. Furthermore the system asks to choose a temporal interval for the publication dates in order to reduce the queried literature; then the resulting query is built up and sent to PubMed. Once the ENTREZ Utilities web service returns the list of articles matching the query (in term of their unique identification number - PID), the user is asked to identify the Semantic Types of interest which the concepts extracted from these literature will belong to; afterwards, for each article, the system queries the LM-DB to generate the list of intermediate concepts (*B concepts*) belonging to the Semantic Types selected by the user that are co-cited in literature with the starting concept, within the time span selected.

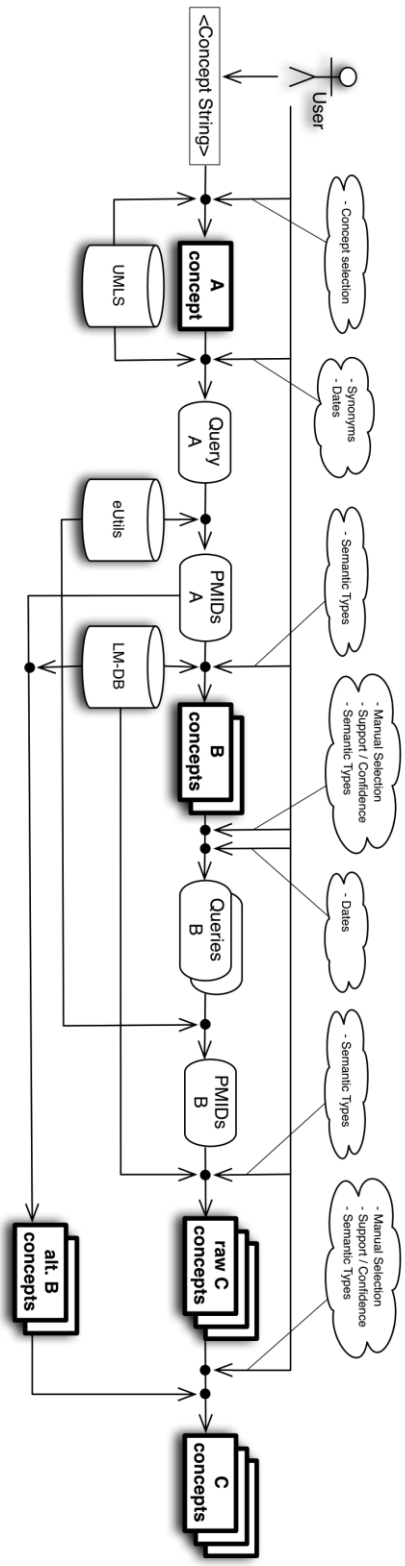

Figure 5.16: The overall workflow of the discovery process underlying the LBD system.

For every B_i concept shown, the system calculates support and confidence of the A→□$_i$ relationship; support is defined as the number of articles where A and B_i are co-cited:

$$Support(A, B_i) = \left\| L_A \cap L_{B_i} \right\|,$$

where L_A and L_{B_i} are respectively the literatures that contain A and a B_i concept; confidence is defined as the relative number of articles where A and B_i are co-cited on the whole literature relative to A:

$$Confidence(A, B_i) = \frac{\left\| L_A \cap L_{B_i} \right\|}{\| L_A \|}.$$

Now the user can manually select one or more B concepts, set up a threshold for support and confidence or apply a new filter on the Semantic Types in order to reduce the set of B concepts that will be used in the final step of the discovery process.

For each intermediate concept selected, the system performs the same steps that have been described before the generation of the set of B concepts from the starting concept A. It is important to notice that also this time the user is asked to define the filtering criteria (publication time interval and Semantic Types) that will be applied in the generation process of the final concepts (*C concepts*) and that these criteria may be different from the ones used in the previous step; this operation may lead to an ambiguity in the discovery process that will be solved in the very final step performed by the system.

Therefore each B concept generates a single query (without the manual addition of the UMLS synonyms) that is sent to PubMed to obtain a list of PIDs. All these lists are then joined together and used to query the LM-DB in order to achieve the set of UMLS concepts that are cited in the whole literature relative to the intermediate concepts (given the filtering criteria defined by the user).

Similarly to what happened with the B concepts, also for these concepts the user can apply some filters: manual selection of single concepts or Semantic Types and set up of a threshold for support and confidence. Since for the B→C step there is not a single concept to start from (B is a set of intermediate concepts), the definition of support and confidence has to be adapted to the case:

$$Support(\hat{B}, C_k) = \left\| \bigcup_i (L_{B_i} \cap L_{C_k}) \right\|,$$

$$Confidence(\hat{B}, C_k) = \frac{\left\| \cup_i (L_{B_i} \cap L_{C_k}) \right\|}{\left\| L_{C_k} \right\|},$$

where $\hat{B}$ is the whole set of B concepts, B_i and C_k are respectively single B and C concepts and $\bigcup_i(\cdot)$ is the union operator.

After this filtering process the system returns to the user a set of C concepts called *raw C concepts*; in fact this set could still contain some concepts that are directly associated with A and therefore they don't represent new associations. In order to delete these concepts, the system could use the list of B concepts if and only if the filtering criteria used in the B→C step (publication time interval and Semantic Types) are the same used for the A→□ step.

Let us explain this with an example: assume our A concept to be a disease, not to filter on publication dates and search for intermediate concepts that still represent a disease (*Semantic Type T047 - Disease or Syndrome*). After the A→□ step the list of B concepts contains a set of diseases that are co-cited in literature with the starting concept; now, for the B→C step, let assume that publication dates of evaluated articles are still not important, and that we now want the LBD system to search for genes (*Semantic Type T028 - Gene or Genome*). The raw C concepts set will then be composed of genes and we have no assurance that these genes are never co-cited with the A concept; it is needed to delete from this set those concepts which are directly associated with A in literature.

If the filtering criteria weren't changed, this set would have been the set of B concepts but, in this case, B concepts are diseases while C concepts are genes: it is needed to determine the alternative B concepts directly associated with A and found with the filtering criteria used in the B→C step. Therefore the same steps that led to the definition of B concepts are repeated with new filtering criteria, the alternative B set (i.e. a set of genes) is defined and, once it is removed from the raw C set, we have the final collection of C concepts guaranteed not to be directly associated with A.

The final set of C concepts, that represent potential new knowledge, is then shown to the user and each concept is associated with its support, confidence and an additional score that takes into account the quality of the intermediate concepts that link it with A; this score is defined as:

$$Score(C_k) = \sum_{i=1}^{N} (Support(A, B_i) * Support(B_i, C_k)),$$

where N is the number of intermediate concepts that link A whit the single final concept C_k. The structure of this heuristic score function guarantees that C concepts that are linked with A by a larger number of links and, in particular, by links with a high support, will achieve a better ranking.

5.3. Case Based Reasoning System

The Case Based Reasoning (CBR) implemented within the present work is designed to deal with flexible case description and allows the comparison of patients coming from different databases where different terms can be used to describe their features set. Such flexibility is needed since:

- cases may come from different databases, so that they could be described by features representing the same concept codified according to different terminologies (e.g. ICD9-CM and SNOMED-CT)
- the features describing the cases may be different as they don't represent exactly the same concepts, but rather similar concepts; for example, a concept may be more general (e.g. "Headache") than another (e.g. "Episodic cluster headache").

The approach implemented allows comparing two cases (by producing their distance) also when data are heterogeneous for codification and detail level.

The main advantage of this approach is its capability to consider the features describing the patient at different levels of abstractions; to accomplish this task, data representation has been founded on biomedical ontologies and terminologies, with particular reference to UMLS, and, consequently, a semantic distance algorithm has been adopted to exploit this data representation.

Our CBR systems can be classified, according to the definitions given in Chapter 4, as an interpretative CBR system; in fact, the main goal of the system is the classification of an incoming case, given its features and a set of prior cases already stored inside the system. The classification is performed on the basis of the comparison of the new case's features with the features of the old cases. The typical steps performed by an interpretative CBR system are:

- Situation assessment: selection, among the whole set of available features, of those that are useful to the system.
- Case retrieval: fetch, among the old cases, of those that are suitable to be compared to the one in exam.
- Case comparison: central step in the execution of the CBR system, consisting in the evaluation of the distances between the case in exam and the old cases in order to determine the most similar cases.
- Case re-using: extension of the set of old cases (case base) with the case in exam to be used for further reasoning processes.

The developed CBR system faces these steps in an unusual way: to date the situation assessment is quite stiff, in fact the system works with cases characterized exclusively by descriptive features; these features, for example symptoms and diagnoses, are those that can be specified by a concept and do not need an associated value, like laboratory and medical

measures, to have sense. In fact, the developed distance measure, that is described in Section 5.3.3, is based on concepts mapped onto a hierarchy and, at the present stage, does not take into account different values that can be associated to a concept.

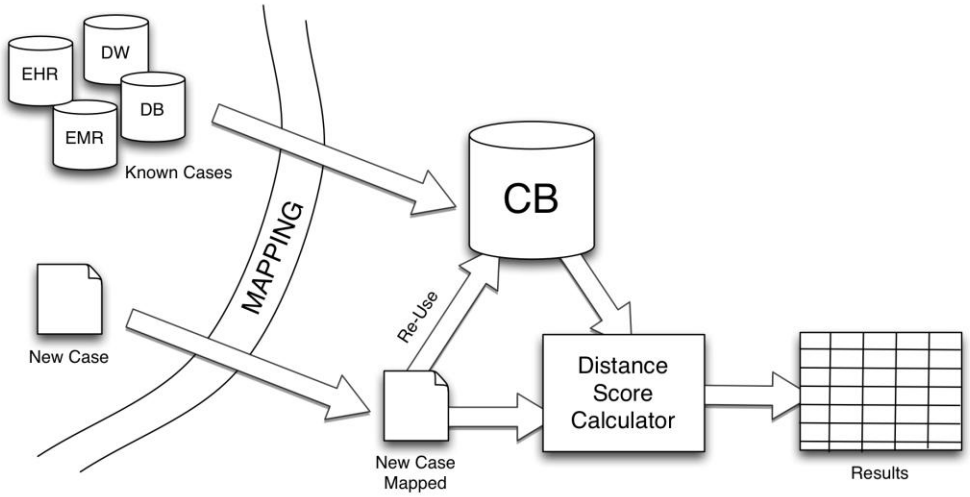

Figure 5.17: Overall architecture of the Case Based Reasoning system.

The very nature of the developed system doesn't require the case retrieval step to be performed; in fact, since the features are mapped on the UMLS Metathesaurus, all the cases are consistent and can be compared. This general purpose case representation technique allows to prevent also the retrieved cases adaptation problem (see Section 4.3.2) because, once a case has been mapped onto the system's internal data representation, it does not need anymore to be adapted. Rather, a process that gets close to adaptation is the mapping of the cases' data onto the common codification environment, but it is an operation done only once, when the case is imported into the case base. It is typically a trans-codification process that can be faced automatically or, at least, semi-automatically. In other words: while the typical adaptation problem forces the CBR systems to adapt, depending on the specific reasoning process, the retrieved cases representation each time the system is used, the developed CBR system forces the cases to adopt its internal UMLS-based data representation, so that the adaptation process is done only in the import phase and consists in assigning to the case's features, usually codified in the original databases, the relative UMLS code. Since the source codification system is typically part of the UMLS Metathesaurus, in most cases the adaptation process is a straightforward operation.

The case comparison step is performed with a semantic distance measure that exploits the same semantic environment (i.e. UMLS) used to represent the data. Finally, the case re-using is very simple because, once the cases have been used and therefore mapped onto the system's internal data

representation, they are ready to be added to the case base for further reasoning processes.

Following the classification given in Section 4.3.4 [137] the system is suitable to be part of the family of Classification Systems; in fact, despite the evaluated features are typically symptoms and diagnoses, the goal of the CBR process is not to directly suggest the diagnosis of the case in exam in order to improve the diagnostic process entrusted to the physician; instead, the developed CBR system computes the distance between the new case and the cases in the case base in order to identify which prove to be similar; then the user can exploit this subset of cases to achieve a better understanding of the current case. In other words, the output of the system is not limited to a single information (or a set of information), but the output is actually given by all the similar cases and their data..

Some aspects of analogical reasoning systems are common to the CBR system: first their cognitive model, that is solving new problems exploiting past experiences, is the same; moreover both systems avoid the case retrieval. Nevertheless, while in analogical reasoning this is a sort of structural restriction, the CBR system can move on this step because of its UMLS-based internal data representation. Anyway many differences exist between these two approaches: analogical reasoning systems do not consider the mapping of the cases onto a space where they can be compared and typically do not re-use the cases taken in exam for future analyses.

The overall architecture of the CBR system is represented in Figure 5.17. The cases of the internal system Case Base (CB) come from several source systems, such as Medical Records, Electronic Health Records, clinical datawarehouses, custom clinical databases and also spreadsheets. These cases, to become part of the CB, have to pass through the process that converts their describing features in the CBR system internal format: this process is called *mapping*. With the CB populated with an adequate set of mapped past cases, the system is ready to accomplish the task of supporting the physician in the reasoning process on a new, still-not-solved, clinical case.

When a new case is submitted to the CBR system, its features pass through the same mapping phase. In this way it is possible to compare the new case with all the cases of the CB and produce, for each comparison, a distance score representing how much the new case is related to each case of the CB on the basis of the features that have been imported in the system. This is the output of the CBR system that can be used by the physician to identify a limited set of cases that are similar to the one in exam and to exploit their data in order to achieve a clearer understanding of the new case.

5.3.1. The Internal Case Representation

The internal case representation of the CBR system is based on the XML format. The features that characterize the single cases and that are suitable

to be introduced in the CB are, to date, the descriptive features that can be mapped onto UMLS and are meaningful per se, without an associated value. Ultimately, the case representation can be seen as an array of features, associated to a UMLS concept (i.e. its Concept Unique Identifier or CUI), and their names that are useful to make the data representation easily human-understandable, although not used to calculate the distance scores.

The features mapping onto the UMLS Metathesaurus arises the problem of selecting the source dictionaries among the many included in the Metathesaurus. The case representation itself would not introduce intrinsic constraints on the concepts source dictionaries; in fact, it can be seen as a simple collection of UMLS concepts. The problem arises when these data have to be used to compute the distance scores between the cases; in fact, as we will explain in Section 5.3.3, not all the relations between the concepts of the Metathesaurus are used to compute these distances, but only those belonging to a set of specific source vocabularies; therefore, in order to be used, also the concepts representing the cases' features have to be mapped on those source vocabularies.

Beyond the CUI and the concept name, another attribute characterizes each feature in the internal case representation: a modifier that indicates if the specific feature has been observed as present or absent in the specific case. This extension to the basic data representation is needed in order to avoid the loss of important medical information as, for instance, those diseases that have been discarded after an investigation. In the case comparison phase the two sets of features, present and absent, will be split and will generate two distinct distance scores; in fact, although both medical concepts, it is not possible to infer a relation between a feature that is present for a patient with a feature that is absent for another.

An example XML of a case belonging to the CB is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<cchart>
  <patient_id>102</patient_id>
  <concept>
    <CUI>C0260662</CUI>
    <status>yes</status>
    <name>hearing disease</name>
  </concept>
  <concept>
    <CUI>C0007682</CUI>
    <status>yes</status>
    <name>CNS disorder</name>
  </concept>
  <concept>
    <CUI>C0014130</CUI>
    <status>no</status>
    <name>endocrine system disease</name>
  </concept>
  <concept>
```

```
      <CUI>C0025362</CUI>
      <status>no</status>
      <name>mental retardation</name>
    </concept>
    <concept>
      <CUI>C0015397</CUI>
      <status>yes</status>
      <name>eye disease</name>
    </concept>
    <concept>
      <CUI>C0263661</CUI>
      <status>no</status>
      <name>disorder of skeletal system</name>
    </concept>
  </cchart>
```

This representation strategy gives to the system a high level of flexibility; in fact cases are not forced to have a declared value for each possible feature and, nevertheless, once they have been split according to present and absent features, they can all be compared with each other.

5.3.2. UMLS Navigation

Both the case representation and the distance score algorithm are based on the UMLS Metathesaurus; for this reason one of the key components of the CBR system is the tool that allows to interrogate and navigate the Metathesaurus: this tool is called *UMLS Navigator*.

The *UMLS Navigator* has been developed in Java programming language and is used, as well as by the CBR system, also by the Literature Mining System when the UMLS concepts extracted from literature have to be characterized with information such as the Semantic Type of the concept, its Type Unique Identifier (TUI), the best ranked source vocabulary and the preferred name in the best ranked vocabulary.

UMLS Navigator must be set up whit some input parameters in order to work properly:

- coordinates and access parameters of the UMLS Metathesaurus database installed on the system. The *UMLS Navigator*, like all the other developed systems, relies on the *Rich Release Format* of the Metathesaurus [172].
- coordinates and access parameters of a database called *cache database*, developed in order to speed up some of the operations performed by the *UMLS Navigator* and that will be detailed later.
- list of source vocabularies on which some of the functions of the *UMLS Navigator* rely.
- four boolean flags that determine which types of relation between UMLS concepts are taken into account by some functions of the *UMLS Navigator*. The considered relations are:

- *PAR* – "has parent (broader hierarchical term)"
- *CHD* – "has child (narrower hierarchical term)"
- *RB* – "has a broader relationship"
- RN – "has narrower relationship"

Table 5.20 summarizes all the functions (i.e. Java methods) provided by the *UMLS Navigator*.

Table 5.20: UMLS Navigator's functions.

Function	Description
findCUI	Returns all the CUIs that can be associated to a given name (three different matching strategies)
findBestName	Returns the prefered name in the best ranked vocabulary relative to a given CUI
findBestNameDictionary	Returns the name of the best ranked dictinary that contains the given CUI
findAlternativeNames	Returns all the names associated with a given CUI in the Metathesaurus
findUpperCUIs	Returns the list of CUIs that are linked to a given CUI with a "PAR"or "RB" relation
findLowerCUIs	Returns the list of CUIs that are linked to a given CUI with a "CHD"or "RN" relation
findSynonimCUIs	Returns the list of CUIs that are linked to a given CUI with a "RQ" relation (related and possibly synonymous)
findSiblingCUIs	Returns the list of CUIs that are linked to a given CUI with a "SIB" relation (has sibling)
findShortestPath	Returns the list of the shortest path that connect two CUIs considering only selected dictionaries
findShortestPathCache	Returns the list of the shortest path that connect two CUIs considering only selected dictionaries and exploiting cache database
findNumberOfDesc	Returns the number of concepts that are recursively connected with a given CUI by relation of type "CHD" or "RN"
findNumberOfDescCache	Returns the number of concepts that are recursively connected with a given CUI by relation of type "CHD" or "RN" exploiting cache database
findTotalConcepts	Returns the total number of concepts that belong to the set of specified source vocabularies
findSemTypes	Returns the Semantic Types and the TUIs associated with a given CUI

The *findCUI* function queries the Metathesaurus to find all the CUIs associated to a given name; there are three different matching strategies for this research task:

- *exact match*: requires the given name to be exactly (with case-insensitive logic) one of the names associated to the CUI.
- *border match*: requires the given name to be part of one of the names associated to the CUI.
- *total flexibility*: requires all the words that compose the given name (separated by blank spaces) to be part of one of the names associated to the CUI.

The output of this function is a list of all those concepts that have at least one name that matches the input string.

The *findBestName* function, provided with an input CUI, returns the preferred name associated to this concept in the best-ranked source vocabulary of the Metathesaurus that contains the CUI.

FindBestNameDictionary, provided with an input CUI, returns the name of the best-ranked source vocabulary that contains the concept.

FindAlternativeNames returns all the names, from all the source vocabularies of the considered UMLS installation, associated with a CUI given in input.

The *findSemTypes* function returns, given an input CUI, all the semantic types and relative TUIs associated with that concept.

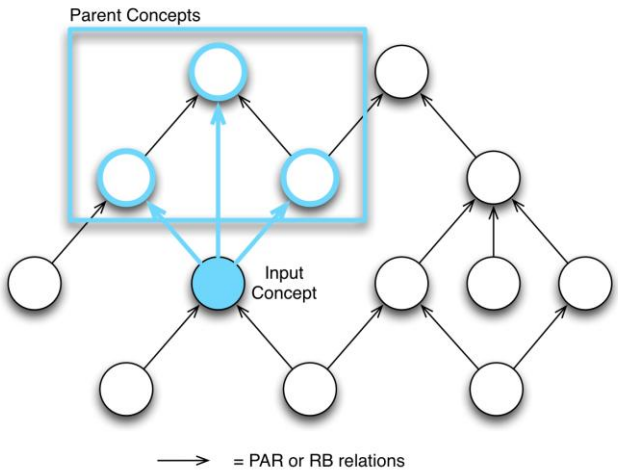

Figure 5.18: Parent concepts in the UMLS Metathesaurus.

These five functions constitute the basic research tools provided by the *UMLS Navigator*, they don't use the *cache database*, nor the list of selected source vocabularies and the four flags associated with the relation types.

The following functions instead work on the subset of the Metathesaurus defined by the selected source vocabularies.

FindUpperCuis is a function that, provided with a specific CUI, queries the Metathesaurus in order to find the concepts of the selected vocabularies directly linked with the one in exam by a "broader" relation; according to the specific boolean flags, these relations can be "PAR", "RB" or both. A graphical exemplification of this process is given in Figure 5.18.

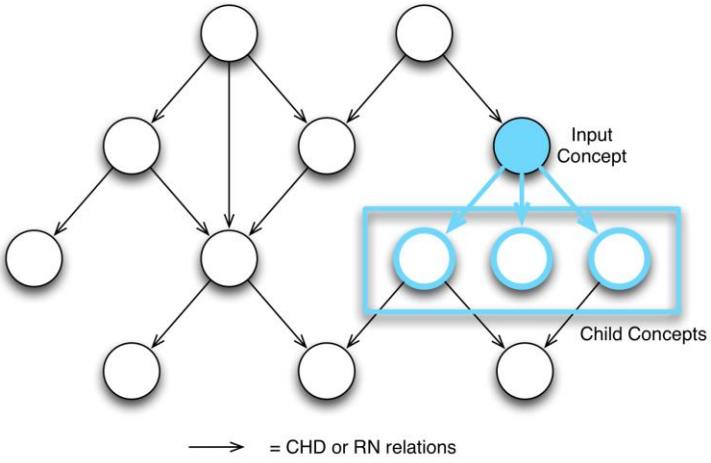

Figure 5.19: Child concepts in the UMLS Metathesaurus.

FindLowerCuis is a function that, provided with a specific CUI, queries the Metathesaurus in order to find the concepts of the selected vocabularies directly linked with the one in exam by a "narrower" relation; according to the specific boolean flags, these relations can be "CHD", "RN" or both. A graphical exemplification of this process is given in Figure 5.19.

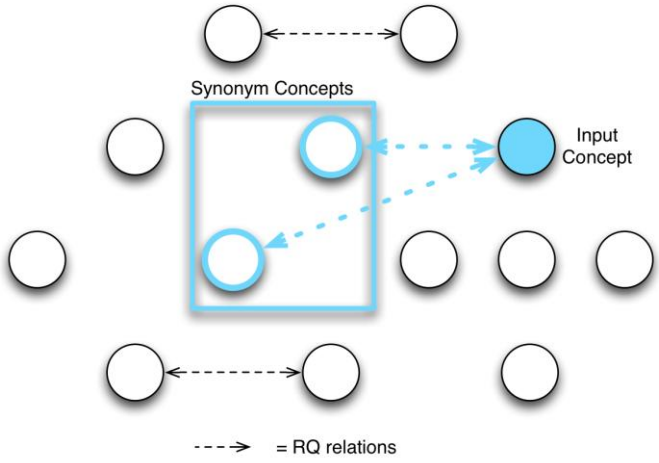

Figure 5.20: Synonym concepts in the UMLS Metathesaurus.

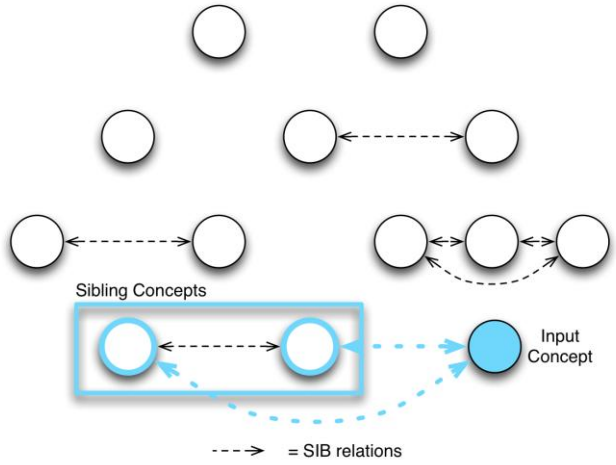

Figure 5.21: Sibling concepts in the UMLS Metathesaurus.

The function *findSynonymCUIs* is similar to the two just described; it returns the concepts that are linked with a "RQ" (related and possibly synonymous) relation to a given CUI (Figure 5.20). Also the function *findSiblingCUIs* follows the same logic, but it considers the "SIB" (has sibling) relations (Figure 5.21).

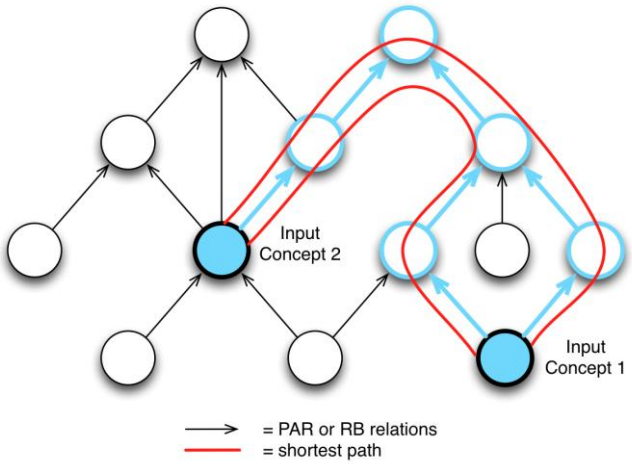

Figure 5.22: Shortest paths between two concepts in the UMLS Metathesaurus

FindShortestPath is a function that, given two input CUIs, recursively scans the selected subset of the Metathesaurus and extracts all the possible paths connecting them. This path search function uses, depending on their boolean flags, the "PAR" and "RB" relations; the algorithm works extending recursively all the paths that start from the input concepts until a common "ancestor" is found. If at least one path joining the two concepts exists, the function returns this path (i.e. the list of intermediate CUIs

between the two concepts) eventually along with other paths of the same size. The working logic of this function avoids infinite loops caused by possible cycles in the considered trees. Graphical examples of how the shortest paths are identified are given in Figure 5.22.

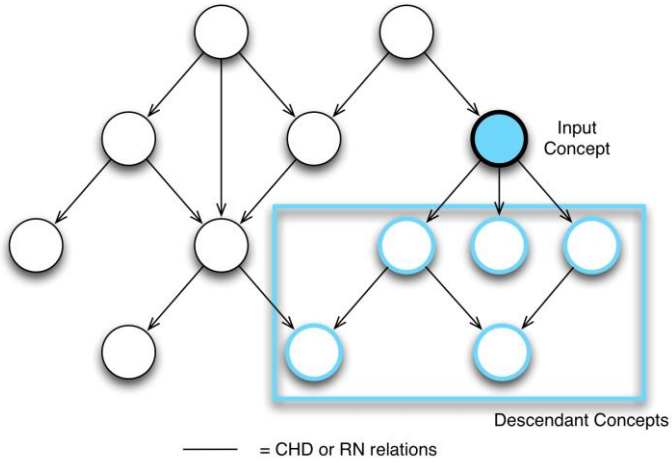

Figure 5.23: Descendant concepts in the UMLS Metathesaurus.

FindNumberOfDesc on the contrary works with "CHD" and "RN" relations (depending on their flags) and returns, given an input CUI, the number of its "descendants" (at all levels) exploring the selected vocabularies, see Figure 5.23.

The last two functions, since their execution can be a very time-consuming task, have been also re-implemented in order to exploit the *cache database* that stores the results of their previous executions. These two functions (*finsShortestPathCache* and *findNumberOfDescCache*), when called, first check in the *cache database* if the results for the same query (i.e. same input CUIs, same selected vocabularies and same relations used) have already been calculated; if it is the case, the function returns the old results (with a considering saving in time and resources), otherwise it calculates the new results, store them in the *cache database* and finally returns them as output.

Also the *findTotalConcepts* function exploits the *cache database* to return the overall number of concepts belonging to this set, considering the selected source vocabularies.

5.3.3. Distance Score

The developed distance algorithm is aimed at providing a tool for comparing heterogeneous cases onto a common codification layer constituted by the UMLS Metathesaurus; a distance score measures the

degree of closeness of compared cases in order to identify analogous cases that can be used at a later stage for disparate clinical and research purposes.

Two distinct approaches are adoptable: the first is a knowledge-free approach based on statistical measures, such as term frequency and co-occurrence [173], but this approach is often non ideal because it disregards the complexity of the medicine domain. Knowledge-full approaches, like the one adopted in the present work, on the other side, exploit the informative-rich framework provided by the many medical terminologies and ontologies to achieve better human-like performance in the comparison of clinical cases.

When the knowledge-full approach to clinical case comparison is adopted, it often happens that small, manually edited databases are used to manage patients data and their comparison inside focused, specific clinical contexts [174] [175]; this approach could be outdone by exploiting the potential of large clinical databases but, in such a much more complex environment, it is not possible to base the comparison of clinical cases on purposely-developed algorithms and a general, flexible comparison mechanism is needed.

The distances obtained are obviously dependent on the information source used for their computation and, when comparing cases coming from different databases, where different terms are used to describe their features set, the best choice was to adopt the most general resource in this field: the UMLS Metathesaurus.

In [176] it is presented a knowledge-full approach to inter-concept distance in UMLS; the authors show a method to automatically calculate a quantitative similarity score between two concepts belonging to the Metathesaurus and, moreover, they investigate the techniques to assess similarity between concepts belonging to a particular UMLS source vocabulary and between concepts generally belonging to the Metathesaurus. Within the present work we extend this UMLS-based knowledge-full approach from an inter-concept to an inter-patient perspective, where the single case (i.e. the patient) is represented as a collection of concepts belonging to the Metathesaurus.

The developed distance metric algorithm is inspired to one of those presented in [177] but, instead of working with concepts belonging to a specific biomedical terminology, SNOMED-CT in that case, it allows concepts to be mapped potentially on the whole Metathesaurus, even if we chose to limit the source vocabularies to a specific set, in order to achieve more reliable results.

The choice of limiting the number of source vocabularies was done in order to avoid potential inconsistent relations between concepts that could be due to the uncontrolled use of all the available sources. In particular, we chose to use the three source vocabularies tested in [176], but, while in [176] they have been used separately, in the present work they were used together to form a unique sub-tree of the Metathesaurus. The three source vocabularies are:

- the Medical Subject Headings standard terminology (MeSH);
- the Systematized Nomenclature of Medicine – Clinical Terms (SNOMED-CT);
- the International Classification of Diseases, ninth version, Clinical Modifications (ICD9-CM).

It is important to note that, anyway, the choice of vocabularies on which the case comparison is founded, is only the default set-up of the CBR system; in fact, since the distance score algorithm exploits the *UMLS* Navigator, it is sufficient to change this tool's list of vocabularies to make the whole CBR system work on a different sub-tree of the Metathesaurus.

The distance score computation is based on the shortest path between concepts belonging to a given interconnected terminology; thus, it is important to select, among the different relation types between two concepts in the Metathesaurus, which are more suitable to be used for our purposes. The two relation types more compliant with the standard "*is a*" relation are:

- *PAR* – "has parent (broader hierarchical term)".
- *RB* – "has a broader relationship".

In [176] both these relations have been tested in a shortest path-based distance measure and the authors concluded that, although both are usable for such purposes, PAR relations are more reliable in finding connecting paths between concepts and, in general, less computationally expensive to use. For such reasons the default behavior of the CBR system is to use exclusively PAR links between concepts, but it is possible to change this by specifying a different choice in the *UMLS Navigator*.

After the mapping phase, the data are represented as an array of concepts belonging to the UMLS Metathesaurus; each concept is represented by its Concept Unique Identifier (CUI) and by its boolean modifier, which indicates if the finding referring to the concept is asserted (present) or negated (absent). One of the main advantages of this approach stands in the fact that the achieved distance scores exploit the same semantic environment used to map the data. The distance score is calculated between two cases, P_N (the new patient) and P_x (a patient from the CB); the generic representation of a case is:

$$P_x = \{feature_1, feature_2, feature_3, \dots, feature_N\}$$

where $feature_i$ is the i-th feature (asserted or negated) associated with the case.

The system calculates the distance score $Dist()$ between P_N and as follows:

$$Dist(P_N, P_x) = w^+ \cdot semDist(P_N^+, P_x^+) + w^- \cdot semDist(P_N^-, P_x^-)$$

where P_N^+ e P_x^+ are sub-arrays of P_N and P_x containing the CUIs of the concepts asserted (i.e. findings that are asserted in the case description), while P_N^- and $P_x^{-\cdot}$ are the sub-arrays for the negated CUIs. w^+ and w^- are balancing factors obtained as follows:

$$w^+ = \frac{size(P_N^+)}{size(P_N^+) + size(P_N^-)}$$

$$w^- = \frac{size(P_N^-)}{size(P_N^+) + size(P_N^-)} = 1 - w^+$$

$Dist()$ returns 0 when two identical cases are compared. The function $semDist()$ is the semantic distance between two arrays of CUIs; it is calculated on the basis of one of the metrics presented in [177]:

$$semDist(P_A, P_B) = \frac{\sum_i min_j(clinDist(P_{A_i}, P_{B_j}))}{size(P_A)}$$

where P_A and P_B are arrays of CUIs, P_{A_i} is the i-th feature of P_A (i.e. a CUI) and P_{B_j} is the j-th component of P_B, and the clinical distance $clinDist()$ between two CUIs is the size of the shortest path joining them in UMLS source vocabularies considered, calculated with the *UMLS Navigator* tool as described in Section 5.3.2. When no path connects the two concepts, the system assigns an arbitrary high clinical distance.

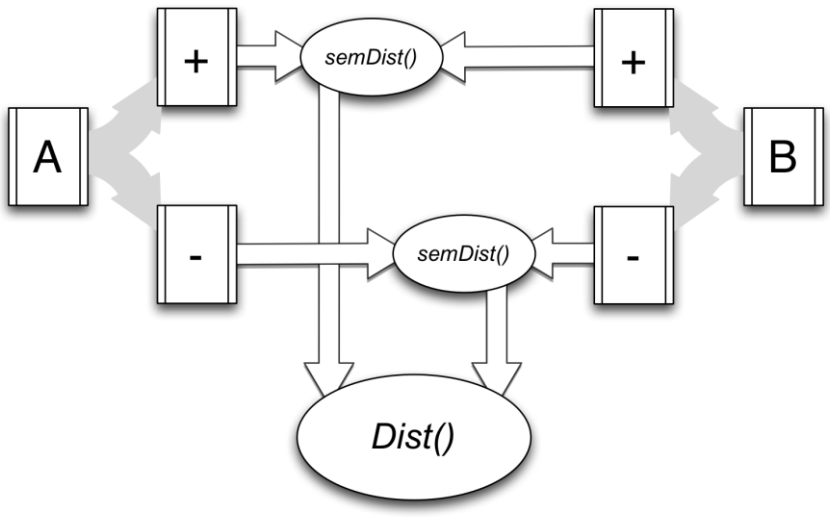

Figure 5.24: Workflow of the distance score algorithm.

Once the distance scores between the incoming case and the cases from the CB have been calculated, the user can visualize and explore the data

about the most similar cases, in order to acquire potentially useful information for the diagnosis/treatment of the incoming patient.

The choice of considering this specific distance algorithm and not, for instance, one of the more sophisticated ones presented in [177], is basically due to the conclusions given by the authors about their experience; in fact, among the algorithms they have tested there was virtually no difference in the obtained results (i.e. the metrics achieved comparable results) and moreover, since we extended this approach to be used potentially with the whole Metathesaurus, the choice of the less computationally expensive one was more advisable.

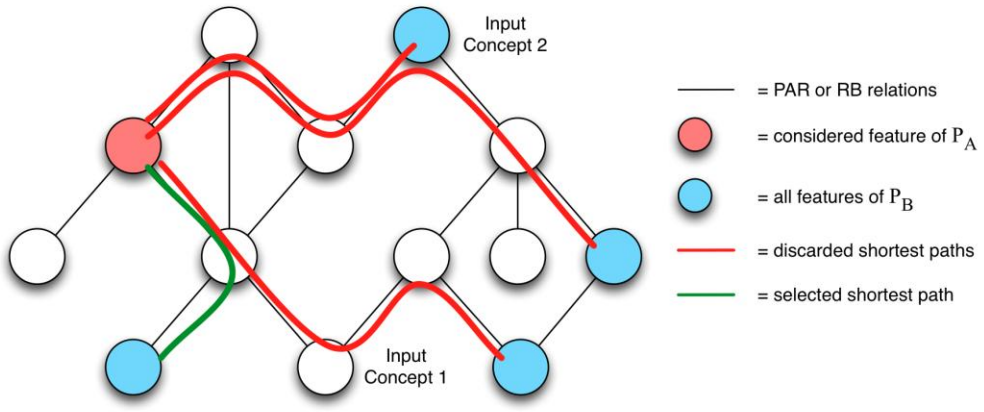

Figure 5.25: Selection of the shortest path to contribute to the overall distance score.

Also the distance score algorithm has been implemented with Java programming language; when two cases, described by their features (asserted or negated) in the internal system representation (i.e. the described XML file), are submitted in order to achieve their distance score, the system first splits their features respectively in two separated sub-sets, one for the asserted features and one for the negated, that will be compared separately and which distance scores will be added up at the end (see Figure 5.24). For each sub-set achieved, the system considers in turn the features of the first case (i.e. P_A in $semDist(P_A, P_B)$) and, for each feature, calculates the shortest paths to all the features of the second case (i.e. P_B in $semDist(P_A, P_B)$); among all these paths only the shortest one ($min_j(clinDist(P_{A_i}, P_{B_j}))$ contributes to $semDist()$ (see Figure 5.25). When all the features of the first term have been considered and the respective shortest paths' lengths have been summed, the system divides the achieved result by the number of features of the first term: the semantic distance between the two sub-sets of features is calculated. After this process is carried on for the asserted features, it is repeated for the negated features and the two partial distance scores are combined (through a weighted sum

as described before) in order to achieve the global score of closeness between the two cases.

5.3.4. From Patients to Literature

Since the internal case representation is founded on UMLS, the system allows to easily move from the patient domain, where the features characterizing the cases can be explored and the cases can be compared with each other, to the literature domain, where it is possible navigate the available scientific literature and the structured information extracted and stored in the Literature Mining Database (LM-DB).

The main idea beneath this process it the automatic creation of a query to be submitted to PubMed from the features characterizing the patient; moreover, since these features are UMLS concepts, this query is far from the simple logical AND of the feature names, but, for example, it can include also the alternative names associated to the concepts.

With an interactive process (described in Section 5.4) the user can also specify the query criteria, the temporal interval of publication for the articles, the alternative names to associate to each concept and also to add concepts that are connected with the patient's features, by navigating the Metathesaurus with the *UMLS Navigator*.

Once the query is completed, it can be submitted to PubMed thanks to the ENTREZ Utilities and the achieved articles can then be directly accessed or, in alternative, the data extracted from them by the Literature Mining System can be exploited. The last function, although accessible in the Java development environment, has not been yet implemented in the Graphical User Interface of the system.

5.4. Graphical User Interface

Alongside of the Literature Mining and the Case Based Reasoning systems also a Graphical User Interface (GUI) has been developed to make most of their functionalities accessible from a unique application. The GUI has been developed with the Google Web Toolkit (GWT) (described in Section 5.1.4) as an AJAX web application that exploits the graphical libraries provided within GWT and the *GWT RPC framework* to manage the client-server interactions. The client is only entrusted with the execution of the simple graphical task of keeping the application interface up-to-date with the data coming from the server. On the other hand, on the server side, all the processes related to the execution of the Literature Mining and the CBR systems are carried out. In particular, the functionalities accessible from the GUI are:

- simple PubMed query execution
- advanced PubMed query execution

- literature navigation
- direct link to PubMed web site
- Case Base navigation
- patient case exploration
- distance score computation
- automatic query creation from a patient case

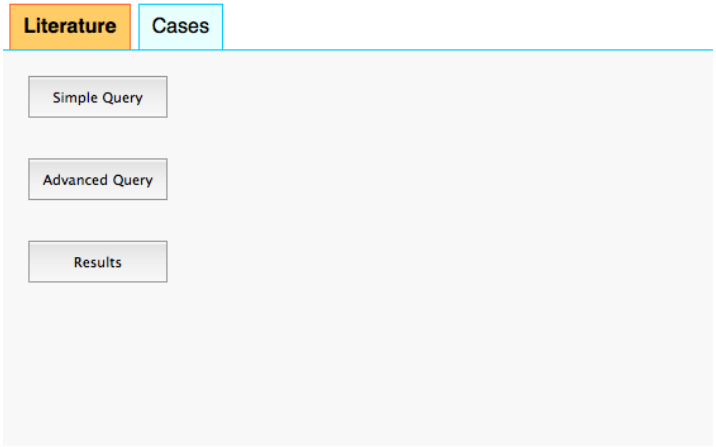

Figure **5.26**: The starting page of the literature navigation system.

The GUI, deployed as a servlet on a standard Apache Tomcat web server [178], can be divided in two sub-systems: the literature navigation system that provides access to the first four functionalities and the case navigation system that supplies the latter four. The user during the execution of the GUI can easily move from on sub-system to the other one with the tab on the right-top of the application window (Figure 5.26).

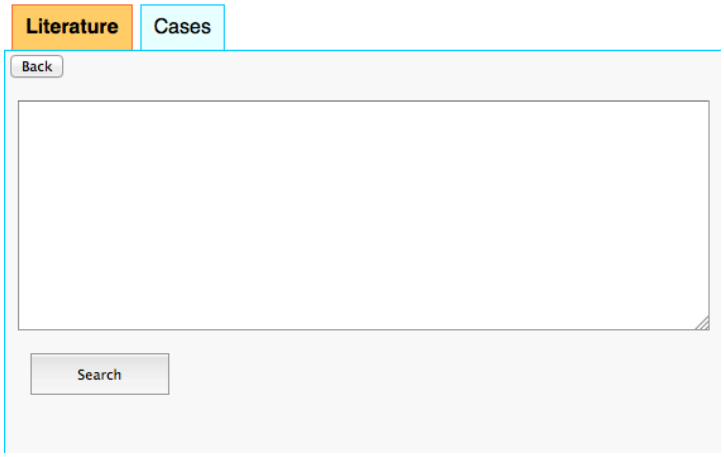

Figure 5.27: The simple query tool.

5.4.1. Literature Navigation System

The starting page of the literature navigation system (Figure 5.26) shows three buttons in order to access the simple and the advanced PubMed query execution and the visualization of the queried literature set. The simple query tool (Figure 5.27) allows the user to manually submit a query to PubMed (via ENTREZ) and then to populate the literature set with the retrieved articles.

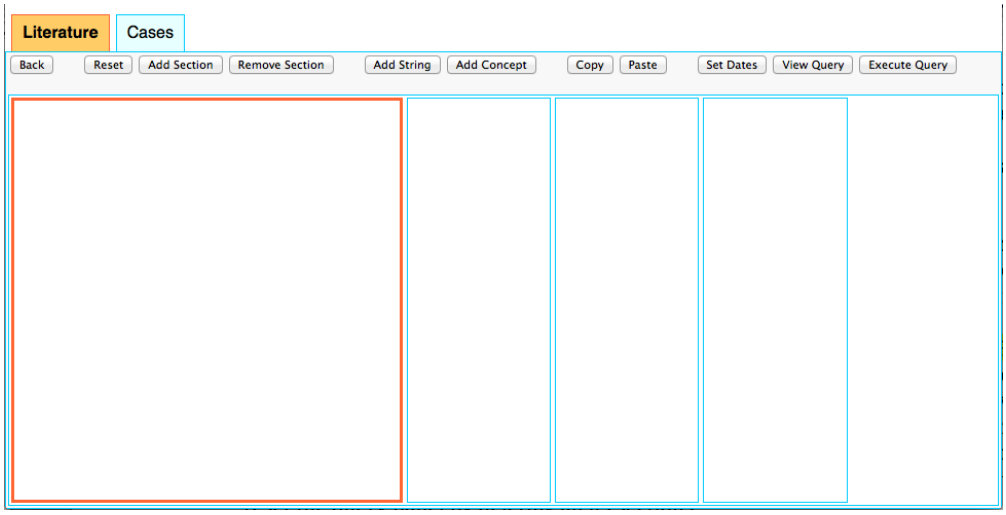

Figure 5.28: The advanced query tool.

The advanced query tool allows the user to create a query with the assistance of the information contained in the UMLS Metathesaurus; the main window of this tool is composed by a toolbar, that displays the functions provided to the user, and the main panel showing the query (*query panel*) (Figure 5.28). The query created with the advanced query tool is a collection of elements that can be either UMLS concepts or text strings; these elements are grouped into sections (the boxes in the main panel); in order to create the query, all the elements of a section are put in logical AND between each other and each section that contains at least one element is put in logical OR with the other sections. At a given time only one section, represented with the orange box, is active and ready to be modified, while the others are inactive; a click of the mouse on an inactive section allows its activation. In the toolbar there are three buttons to:

- reset the query panel by deleting all its sections ("*Reset*")
- add a new empty section to the query panel ("*Add Section*")
- remove the selected section from the query panel ("*Remove Section*")

As remarked, elements that populate a section can be textual strings or UMLS concepts; to add these types of elements there are two specific buttons on the toolbar: "*Add String*" and "*Add Concept*". The string adding

process is simple and requires the user only to insert the desired text and to submit it to the active section (Figure 5.29).

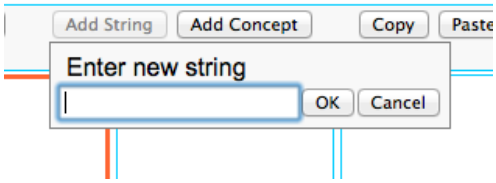

Figure 5.29: The simple process to add a string to the selected section.

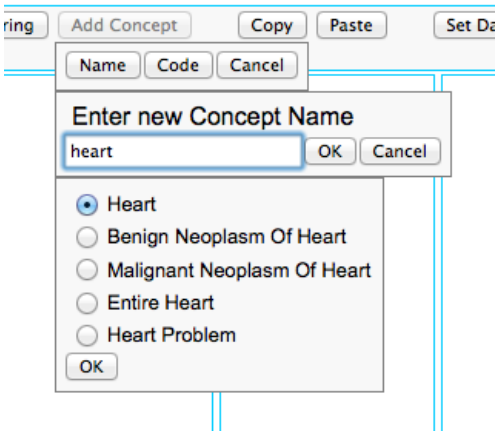

Figure 5.30: The selection process of the UMLS concept to add to the selected section.

A UMLS concept, to be added to the query panel, can be specified either with one of its names or with its Concept Unique Identifier (CUI); when the concept's name is submitted, the system exploits the *UMLS Navigator* and returns all the concepts belonging to Metathesaurus associated with that name, presenting them with the respective preferred name from the best-ranked source vocabulary (Figure 5.30); then the user can select the concept to add to the active section of the query panel.

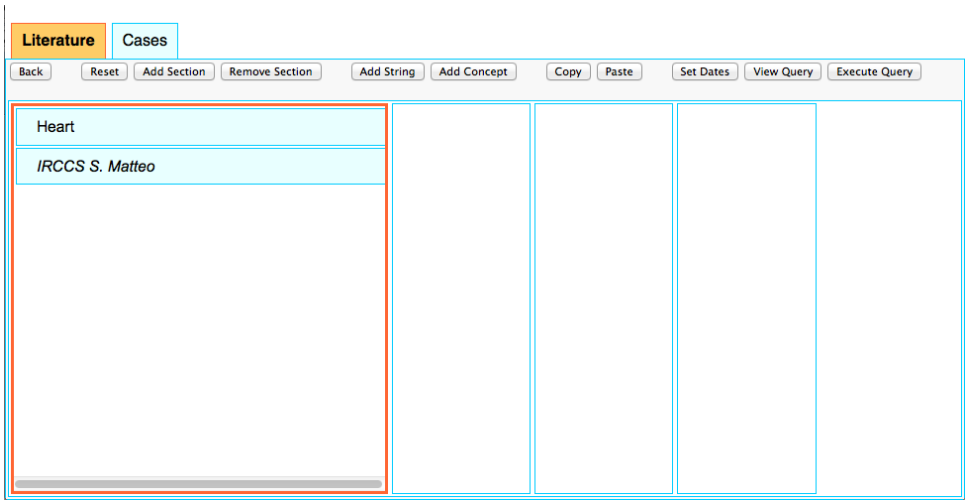

Figure 5.31: The selected section contains a UMLS concept ("Heart") and a string ("IRCCS S. Matteo").

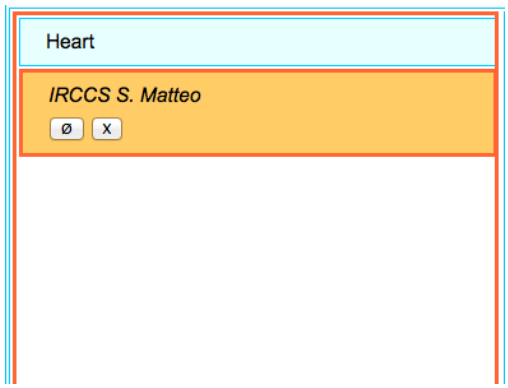

Figure 5.32: Operations allowed on textual strings.

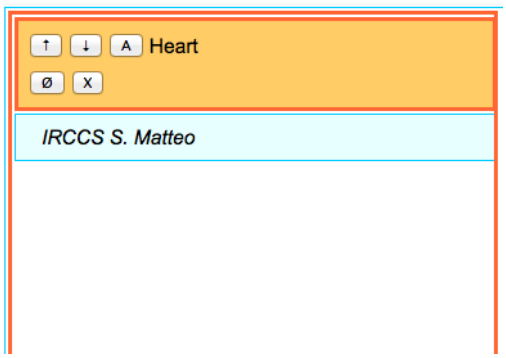

Figure 5.33: Operations allowed on UMLS concepts.

Once an element has been added to a section it appears as a blue box containing the inserted string or the concept's name (Figure 5.31); when an element is clicked by the mouse it expands showing as many buttons as are

the operations that the user can apply on the element. Textual string elements only allow the be inactivated/activated or deleted (Figure 5.32); when an element is inactivated it changes its color to gray and, even if it remains part of the section and can be reactivated, it does not take part to the query.

UMLS concept elements (Figure 5.33), beyond the operations of the textual element, also allow the user to exploit some functions of the *UMLS Navigator*:

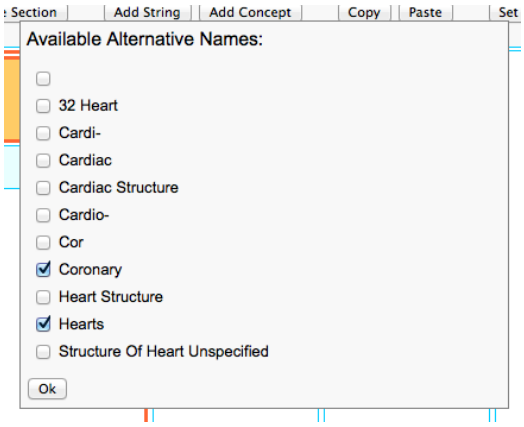

Figure 5.34: Selection of the alternative names to be used in the query.

- to select, among all the names associated to the concept in the Metathesaurus, the names that will be part of the query. Therefore, an element contributes to the query with its preferred name in logical OR with all the selected alternative names (Figure 5.34).

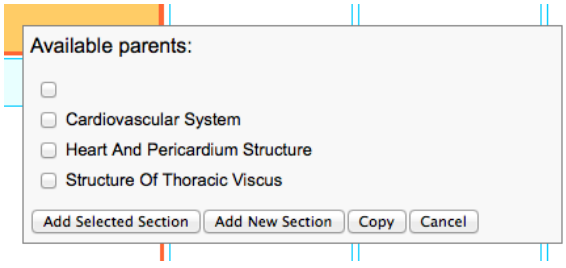

Figure 5.35: Selection of the available parent concepts to be added to the query.

- to add (to the selected section or to a new section) a "parent" concept that the user can select from a list. The parent concepts are those directly linked to the concept with a "PAR" or "RB" relation (Figure 5.35).
- to add (to the selected section or to a new section) a "child" concept selected by the user from a list. The child concepts are those

directly linked to the concept with a "CHD" or "RN" relation (Figure 5.36).

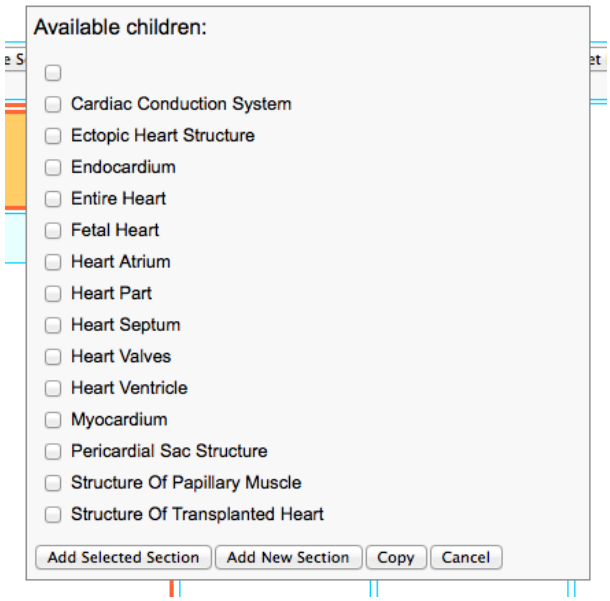

Figure 5.36: Selection of the available child concepts to be added to the query.

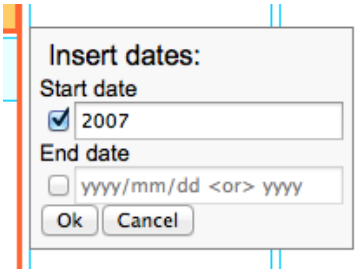

Figure 5.37: Set up of the query's dates

Elements and entire sections can also be copied and pasted with two buttons in the toolbar ("*Copy*" and "*Paste*"). Moreover the user can set up the start and end dates that determine the temporal interval of the articles to be retrieved (Figure 5.37).

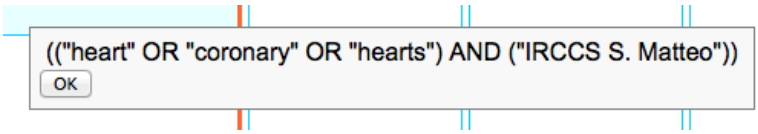

Figure 5.38: The generated query in textual form.

Once all the sections have been set up, the query is ready to be submitted to PubMed. Before launching a query, the user can see it in textual form (Figure 5.38) with the "*View Query*" button; finally, the query can be fired with the "*Execute Query*" button.

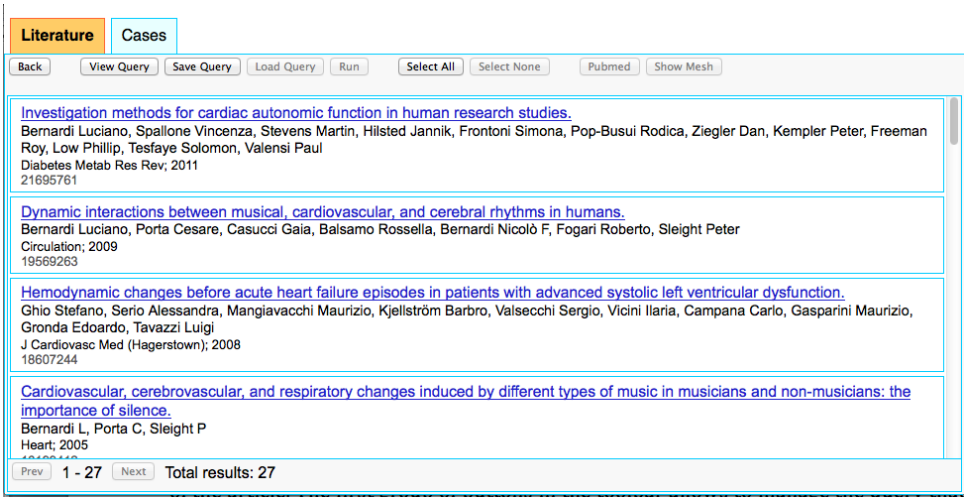

Figure 5.39: PubMed results tool.

When a query, both from the simple and the advanced query execution tools, is executed, the application visualizes the PubMed results tool (Figure 5.39). This tool is composed by a toolbar on the top with the operations available, a main panel that contains the retrieved articles' information and a bottom bar to navigate the results. Each article is represented as a box in the main panel and contains: title, authors, journal, year of publication and PubMed ID of the article. The first group of buttons in the toolbar allows managing the query that can be viewed, saved, loaded and run.

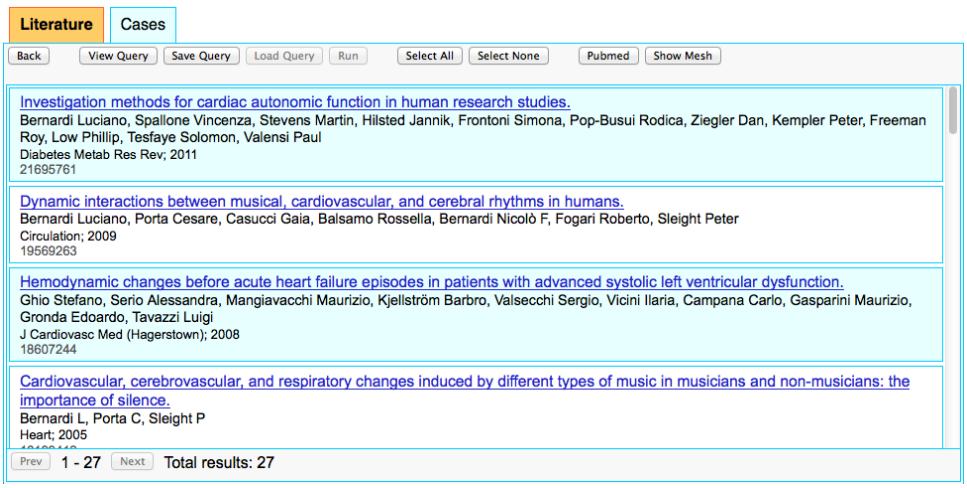

Figure 5.40: Selected articles in the PubMed results tool.

The articles in the main panel can be selected/deselected with a mouse click; when an article is selected, its box changes its color and more than one article can be selected at a time (Figure 5.40). The selected articles set can be exploited directly in the PubMed web site (with the "*Pubmed*" button) or, for each article, the system can obtain the associated MeSH terms and show them in the respective box (with the "*Show Mesh*" button) (Figure 5.41).

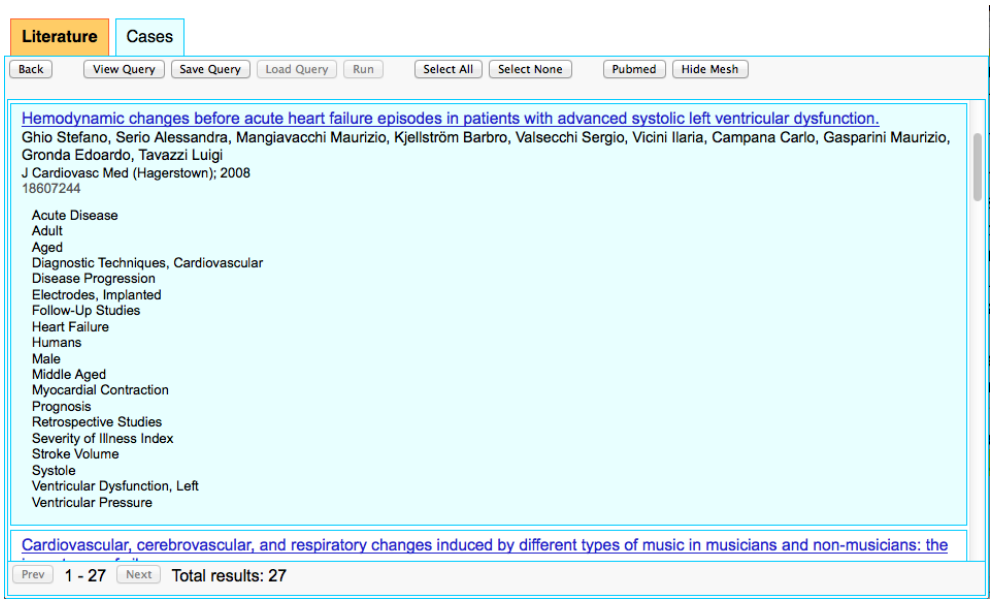

Figure 5.41: MeSH terms relative to an article in the PubMed results tool.

5.4.2. Case Based Reasoning System

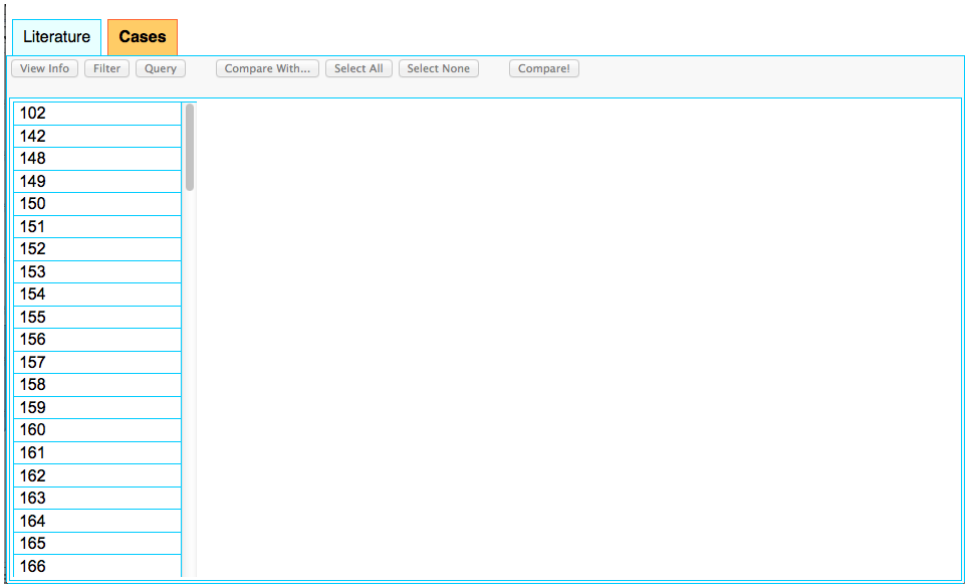

Figure 5.42: The case navigation system's window.

The case navigation system window (Figure 5.42) is composed, on the top, by a toolbar that shows the executable operations, on the left, by a list that contains all the cases loaded in the Case Base (CB) and by a main panel. When one of the cases is selected, two operations become available: "*View Info*" to show in the main panel the features that characterize the selected case and "*Compare With…*" to execute the distance score algorithm.

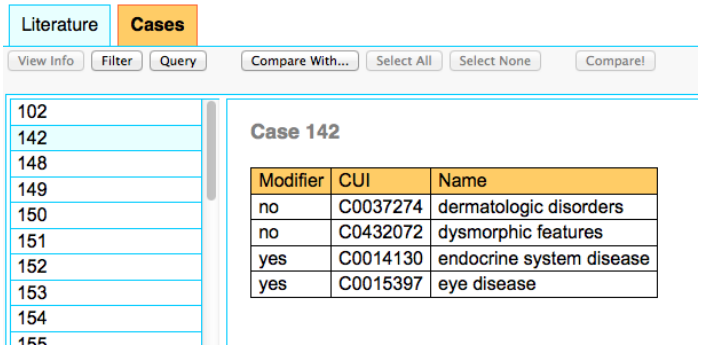

Figure 5.43: Features relative to case 142.

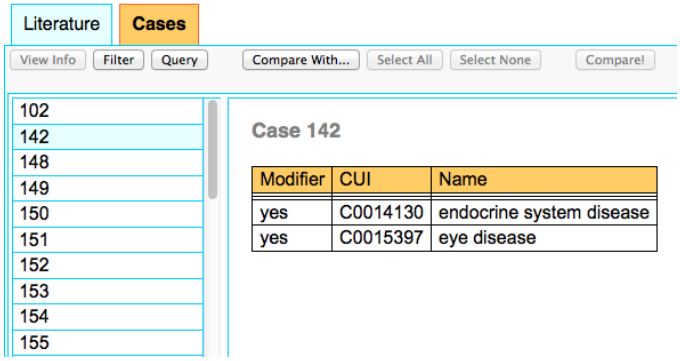

Figure 5.44: Asserted features of case 142.

When the information about a case is shown, a table reporting the case's features appears in the main panel, where each feature is described by its boolean modifier, CUI, and name (Figure 5.43). It is then possible to filter out the data by their boolean modifier (Figure 5.44) or exploit the selected feature set to build up a PubMed query; in this case the system switches to the literature navigation's advanced query tool that is already set up with a section that contains all the concepts from the case's feature set (Figure 5.45); the user can directly trigger the query as it is or modify it with the functions available in the advanced query tool.

When the user launches the case comparison, exploiting the distance score algorithm, the system, beside the patients list, shows all the cases comparable with the selected one (typically all the cases in the CB except the selected case itself); it is then possible to manually select the cases to

compare or, with the "*Select All*" and "*Select None*" buttons of the toolbar, select or deselect all the available cases. When the cases set is defined, the user can start the comparison with the "*Compare!*" button.

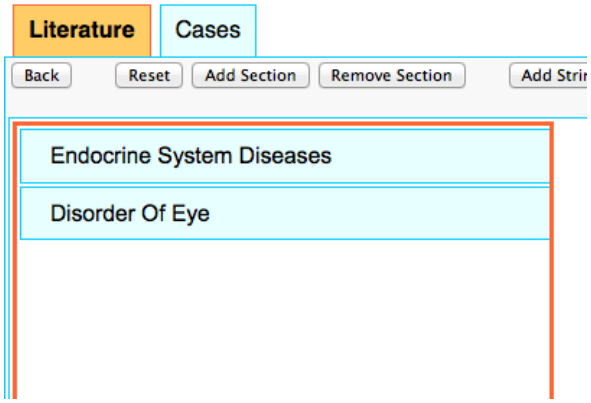

Figure 5.45: The features of a specific case can be used to generate a query to interrogate PubMed.

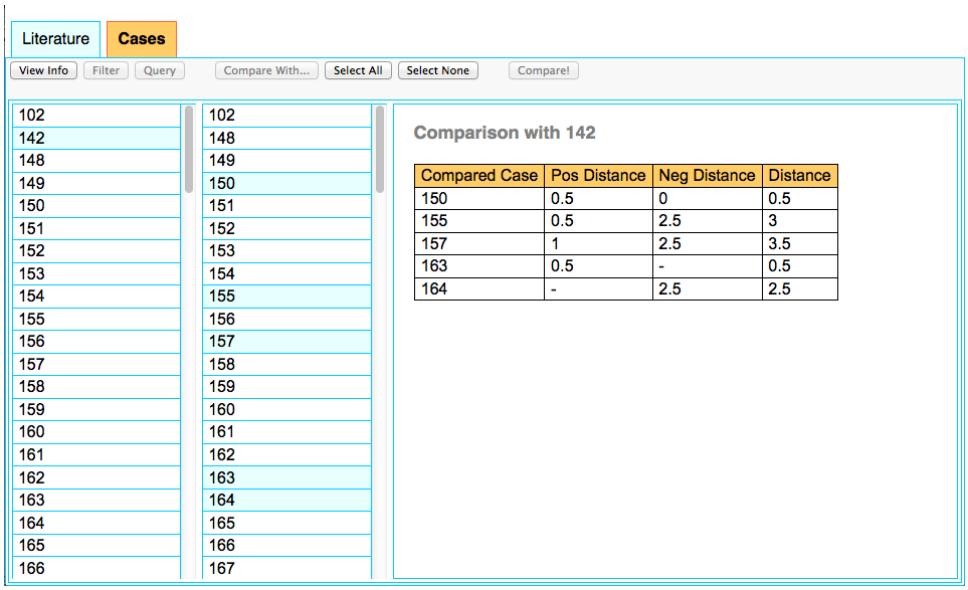

Figure 5.46: Distance scores between case 142 and a set of selected cases.

Once the system has terminated the execution of the distance score computation, it shows the results table in the main panel (Figure 5.46); for each case compared to the one in exam the table shows the distance score between the asserted features ("*Pos Distance*"), the one between the negated features ("*Neg Distance*") and the global distance score ("*Distance*"). By clicking the column headers it is possible to sort the results according to the specific field.

Chapter 6

Applications and Results

In this chapter the contexts of application of the Literature Mining and CBR systems and the achieved results will be discussed; in particular, the two projects taking advantage of the developed technologies will be introduced along with the applications that embed these subsystems.

The first project, called INHERITANCE (Integrated Heart Research In Translational Genetics of Cardiomyopathies in Europe), is a multidisciplinary, multi-center research project funded by the European Commission that seeks to study the genetics of inherited Dilated Cardiomyopathy (DCM) and to understand the impact and management of the condition within families that suffer from DCMs.

The second project is called ONCO-i2b2, an information technology initiative started by the University of Pavia (Italy) and the IRCCS Fondazione Salvatore Maugeri hospital in Pavia to support clinical research in oncology; this project aims at supporting translational research in oncology and exploits the software solutions implemented by the Informatics for Integrating Biology and the Bedside (i2b2) research center [39].

Every part of the two systems has been used in the two projects with the main goal of testing their many possible applications to real-world problems and their adaptability to the specific contexts needs.

The Literature Mining system has been exploited to:

- characterize a set of genes on the basis of the related scientific literature;
- prioritize the genes to be screened for a given patient by evaluating his phenotype features along with the related scientific literature.
- discover the associations between DCM and its causative genes with the aim of evaluating the Literature Based Discovery tool's performance.

The CBR system has been used to:

- assess the distance scores between a set of simulated DCM patients created with the support of physicians involved in the INHERITANCE project.
- extend the ONCO-i2b2 data warehouse's query tolls with the distance score algorithm in order to enhance the patient selection process.

The remaining part of this chapter treats the two projects separately. In Section 6.1 the INHERITANCE project is introduced and the results achieved are presented; Section 6.2 introduces the ONCO-i2b2 project and describes the application of the CBR system to its clinical context.

6.1. The INHERITANCE Project

Cardiomyopathies are defined as primary myocardial disorders of unknown cause and are classified into four main subtypes, based on ventricular morphology and physiology: hypertrophic (HCM), dilated (DCM), restrictive (RCM) and arrhythmogenic right ventricular cardiomyopathy (ARVC) [179]. Overlapping phenotypes (for example, dilatation of a hypertrophied ventricle) can occur in advanced phases of the disease. DCM is defined as a myocardial disorder characterized by the presence of left ventricular dilatation and systolic impairment, in the absence of abnormal loading conditions (e.g. hypertension, valve disease) or coronary artery disease sufficient to cause global systolic dysfunction [179]. Right ventricular dilatation and dysfunction may also be present. Family screening and genetic studies have identified 20 disease-causing genes to date [180].

Currently, patients with DCM are treated in accordance with international guidelines for the management of heart failure with little consideration of the possible influence of the underlying etiology on the response to treatment. Recent studies suggest that this might result in sub-optimal or inappropriate therapy in some patients [181]. For example, knowledge that a DCM patient is carrier of a LMNA gene mutation might be of major importance when deciding on device therapy.

As explained before, the INHERITANCE project seeks to study the genetics of inherited DCM, understand its impact and improve its management within families suffering from DCM. The INHERITANCE translational strategy is based on a clinical algorithm that seeks to determine disease-specific features to be associated with different types of DCM or suggest specific genetic or metabolic pathways of disease. A reverse translational strategy will be run in parallel to establish or confirm the association of DCM phenotypes with clinical markers occurring in DCM patients with different genes mutations.

The INHERITANCE project is structured into six research areas that study different facets of the DCM condition, including clinical cardiogenetics, -omics, i.e. genetic testing, transcriptomics, proteomics and metabolomics, animal studies, structural studies, treatments, and biomedical informatics, which aims to implement information technology solutions to support the project team in managing the huge quantity of scientific, clinical and patient data generated by the project.

The biomedical informatics systems developed for the INHERITANCE project implement a layer of software instruments to support translation of the results of the project into clinical practice as well as to support the scientific discovery process. Among these tools, those exploiting the Literature Mining system (Section 5.2) and the CBR system (Section 5.3) are:

- an automated literature analysis tool that, starting from a set of genes associated with DCM, exploits the Literature Mining DataBase (LM-DB) (Section 5.2.1) to extend this set with new, potentially interesting genes. Afterwards the tool provides a measure of the genes' similarity on the basis of some literature-derived features;
- a reasoning tool that exploits the distance score algorithm described in Section 5.3.3 to cluster the patients of a DCM simulated patient set. The aim is to evaluate the capability of the CBR system in recognizing the relative closeness of patients with similar diagnoses on the basis of their features;
- a tool for gene prioritization that, exploiting the Literature Mining system, can produce, given a specific case, a list of genes to be screened, ranked by a priority score or, in alternative, a general prioritization list for the genes associated to the disease (i.e. DCM).
- a tool for Literature Based Discovery (LBD), based on the analysis of scientific literature. This tool has been tested by simulating the discovery process of DCM causative genes. The association between DCM and potential gene mutations has been achieved by exploiting only the scientific papers published before the first explicit appearance of the association in literature.

The results relative to the biomedical informatics applications developed within the INHERITANCE project are described in [182]. The following sections will describe in detail the tools used and the achieved results.

6.1.1. Literature Analysis

The task of analyzing the relevant scientific literature is particularly important in the early stage of any study, in which resuming the available knowledge is crucial to formulate initial hypotheses and plan next tasks. The challenge is to broaden the search of potentially useful information to generate new hypotheses [183]. For instance, an added value could be to

suggest that a candidate gene is often related to another one, which has not been previously considered. With the INHERITANCE automated literature analysis tool we focused on genetic studies, in which a set of initial hypotheses of gene-disease association is made on some candidate genes, so that the first step is to explore the recent literature to confirm their possible role in the disease mechanism.

This tool exploits the concepts extracted from scientific literature and stored in the LM-DB; in particular each gene is characterized by the co-cited UMLS concepts. On top of this process we have implemented a similarity metric, based on a relevance measure of the terms associated to each gene. In this way we can derive a graph, in which the nodes connections reflect how tightly related these genes are, in accordance to the available literature.

The whole analysis has been conducted in three phases:

- starting from the set of DCM-causing genes, the literature set relative to each gene was retrieved with the *PubmedFinder* (Section 5.2.2) and, by exploiting the results of the *Gene Finder* (Section 5.2.3.2) previously stored in the LM-DB, from each literature set the co-cited genes were extracted. All the co-cited genes from all the literature sets were merged into a single gene set. In summary, the first step is aimed at extending the set of the DCM-causing genes with possibly related new genes.
- each gene belonging to the extended set of genes was used to query PubMed in order to achieve, for each gene, the set of related scientific articles. For each set of articles (i.e. for each gene) a query to the LM-DB is run to obtain the UMLS concepts reported in their abstracts. After this step, each gene of the extended set is associated to a set of UMLS concepts.
- in the third step, the annotation profiles of each gene (i.e. the associated UMLS concepts) are used to build graphical and quantitative representations of the gene network.

In particular, the INHERITANCE literature analysis tool has been used as follows: starting from 20 candidate genes known to be involved in DCM, we applied this strategy with the final aim of identifying a set of DCM-related genes to be further investigated.

The 50 most recent abstracts, obtained by querying PubMed for the candidate genes were used to extract other 2414 co-cited genes and their most frequently associated terms. This procedure allowed to represent each gene by an annotation profile, composed of UMLS terms indicating diseases or symptoms and the counts of their occurrences in PubMed entries. The annotation profiles were composed, on average, of a set of 60 terms and their corresponding frequencies. Thanks to a text mining weighting scheme, known as TF-IDF [184], the counts were transformed into values that reflect the importance of each term for each gene. Within these methods, a term is rated as important for a gene if it occurs frequently

in its articles and if it is specifically assigned to the publications related to that gene.

In order to have a graphical visualization of the groups of similarly annotated genes, we have created an association network, where the genes are linked on the basis of the cosine similarity of their annotation profile.

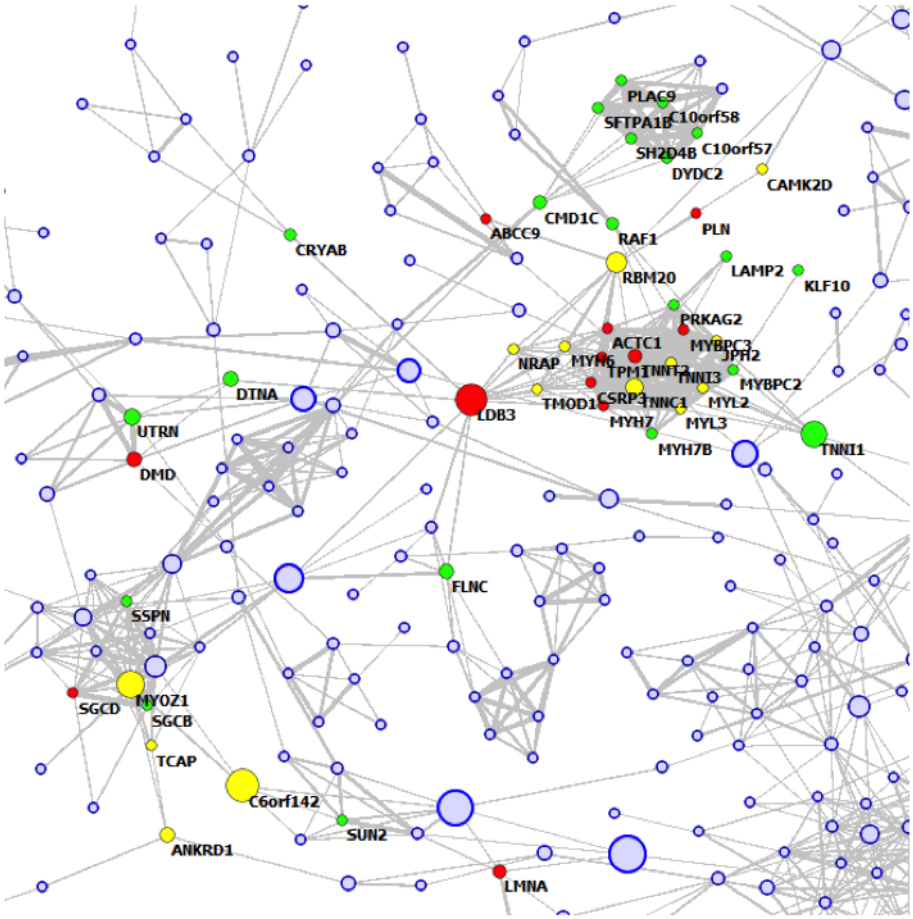

Figure 6.1: Zoom on the DCM network. Red nodes are the initially considered DCM-related genes, yellow nodes are novel genes specifically associated with DCM in the most recent literature, while green nodes represent genes highly co-cited with the term "Cardiomyopathy".

Figure 6.1 shows a portion of the network obtained by connecting the genes if the cosine metric exceeded the 99th percentile of the distribution of pair-wise similarities. Filled-colored nodes in the network represent genes that may be taken into consideration from researchers interested in DCM. In particular, we have selected the 5 most important terms for each gene, according to TF-IDF, and used them to highlight additional genes tightly associated with DCM in the literature (in yellow) and other genes related to the more general UMLS term "Cardiomyopathies" (in green).

To quantitatively characterize the importance of the nodes in our network, we have also computed the betweenness centrality [185]. This

topological measure is defined as the number of shortest paths that go though a considered node, and represents the influence of that node in the flow of information in the network. Nodes with a high betweenness typically make possible the communications in the network among clusters of nodes characterized by high internal connectivity. We observed these properties also in the DCM network, where the size of the nodes was adjusted proportionally to the value of the node's betweenness. For instance, by analyzing the important associated terms, we noticed that the known DCM-related gene LDB3, characterized by a high betweenness centrality, links a group of nodes that are strongly associated with Cardiomyopathies (in the upper right side of the image) to a cluster (in the left side), which is more frequently related to Myopathy and Muscular dystrophy.

6.1.2. Case Based Reasoning

The preliminary testing process of the CBR system (Section 5.3) has been performed on a simulated set of cases. We chose to adopt this strategy in order to be able to measure the CBR system performance also in the early stages of the INHERITANCE project, when the data warehouse was still in development and the data weren't already stored into the project database.

Working side by side with the physicians involved in the project, we created a simulated benchmark of DCM patients in order to test the capability of the CBR system to associate a single case with the most similar cases in terms of actual diagnosis.

The simulated patient set has been designed considering the main known relationships between the different types of DCMs and their typical phenotype markers [186]. In particular, for each class of interest, we derived from literature the event rates of all the features suitable to describe the patient; in this step we have also exploited the experiences of the physicians involved in the project. Afterward, we randomly generated each simulated case considering the probabilities of each feature to be true according to the class it belongs to (see Table 6.1).

We defined four classes of patients:

- not affected by cardiomyopathy (*no CM*)
- affected by a not-specified cardiomyopathy (*CM*)
- affected by DCM with Dystrophin (*DCM dys*) mutation
- affected by DCM with Lamin (*DCM lmna*) mutation

Each patient is described by sixteen phenotype features, related to diagnoses and symptoms (e.g. "Chest Pain", "Dyspnea", "Muscular Dystrophy"); each phenotype feature has the asserted/negated boolean modifier associated. For each class we generated 25 cases that were obviously already in the CB format (Section 5.3.1), so no mapping was needed.

The UMLS version used for testing the system is UMLS-2010AB, while the vocabularies selected for the internal data representation are:

- Medical Subject Headings (MeSH)
- UMLS Metathesaurus (MTH)
- SNOMED Clinical Terms (SNOMED CT)

For the purposes of this test, it was enough to structure the patient description into a single section containing all the features.

Table 6.1: Probability for a simulated patient, belonging to one of the four classes (*no CM*, *CM*, *DCM dys*, *DCM lmna*), to have the relative feature asserted.

Feature	CUI	%			
		no CM	CM	DCM dys	DCM lmna
Cigarette smoker	C0337667	0.25	0.25	0.25	0.25
Diabetes Mellitus	C0011849	0.3	0.3	0.3	0.3
Hypertensive disease	C0020538	0.3	0.3	0.3	0.3
Chest Pain	C0008031	0.1	0.7	0.7	0.7
Dyspnea	C0013404	0.1	0.7	0.7	0.7
Syncope	C0039070	0.1	0.7	0.7	0.7
Muscular Dystrophies	C0026850	0.1	0.1	0.7	0.1
Atrioventricular Block	C0004245	0.05	0.05	0.05	0.7
Left Ventricular Hypertrophy	C0149721	0.05	0.7	0.7	0.7
Inverted T wave	C0520888	0.05	0.85	0.85	0.85
Left ventricilar dilatation	C0344911	0.05	0.05	0.8	0.8
Left ventricular ejection fraction	C0428772	0.05	0.7	0.7	0.7
Increased creatine kinase level	C2316757	0.05	0.05	0.7	0.9
Heart Transplantation	C0018823	0.005	0.3	0.5	0.6
Electric Countershok	C0013778	0.05	0.3	0.3	0.8
Sudden Death	C0011071	0.05	0.7	0.7	0.7

The test has been performed as follows: 3 patients per class (12 patients in the aggregate) were randomly extracted from the original set of 100 patients; in turn, these patients were considered as the new case and the CBR system evaluated their distance score from the remaining 99 cases. The results are shown in Table 6.2.

From the data shown in this table, it is clear that the distance score algorithm works properly; in fact 11 of the considered 12 cases are correctly classified and, in general, each case has a lower distance from the cases of the same class than from the cases of the other classes.

Table 6.2: Distance scores achieved in the test on the simulated patients' set. For each class of the compared cases, the scores are computed as the mean values of the similarities between the case under examination and all the compared cases belonging to the same class. Scores represented vertically are the mean values for the class they belong to. Scores in bold are the best for each new case or class.

			Compared cases							
			no CM		*CM*		*DCM dys*		*DCM lmna*	
Cases in exam	no CM	#1	***2,28***	***3,26***	*7,95*	*8,04*	*9,15*	*9,88*	*10,45*	*10,73*
		#2		***2,26***		*8,05*		*8,3*		*10,01*
		#3		***1,32***		*7,75*		*9,26*		*10,61*
	CM	#1	*9,95*	*9,07*	***3,51***	***4,5***	*4,81*	*5,1*	*5,19*	*5,9*
		#2		*10,33*		***2,88***		*4,29*		*4,48*
		#3		*10,46*		***3,16***		*5,04*		*5,17*
	DCM dys	#1	*11,63*	*10,97*	*6,3*	*7,58*	***4,89***	***6,16***	*5,02*	*6,59*
		#2		*10,52*		*5,96*		***4,74***		*5,16*
		#3		*13,4*		*5,36*		*3,78*		***3,32***
	DCM lmna	#1	*13,26*	*14,03*	*5,21*	*5,43*	*4,34*	*4,26*	***3,04***	***2,79***
		#2		*13,53*		*5,41*		*4,15*		***2,8***
		#3		*12,22*		*4,81*		*4,61*		***3,53***

6.1.3. Gene Prioritization

The Literature Mining system has been used also for candidate gene prioritization; the process to obtain a ranked set of candidate genes exploits the LM-DB and, in particular, the annotations produced by the *Gene Finder* and the *UMLS Finder* (Section 5.2.3.5). This analysis can be performed in two different ways: the first one is driven by a specific real patient case, while the second one is aimed at updating the knowledge about the main disease (DCM) and its comorbidities.

The first analysis is patient-specific and, therefore, is repeated for every investigation:

- for each finding asserted in the current case, PubMed is queried in order to obtain the reference to the directly related articles. Since all the data associated with the patients have been mapped onto the UMLS Metathesaurus, the search can be optionally extended also to the articles related to concepts that are semantically similar to the one present in the patient description. Following this analysis, the system generates a list of the associated genes along with the co-occurrence

frequencies. These frequencies are used to obtain the ranked list of candidate genes for the specific patient.

The second type of analysis includes the following steps:

- by considering the MeSH thesaurus, 17 disease categories are taken into account. These MeSH terms are direct descendant of the "Diseases" main term and represent the diseases macro-categories (e.g. "Musculoskeletal Diseases", "Nervous System Diseases"). The goal is to identify a set of articles where the "Dilated Cardiomyopathy" main disease is co-cited with diseases belonging to these macro-categories (that represent the comorbidities). The PubMed queries resulting from this approach are therefore composed by "Dilated Cardiomyopathy" (searched within the whole abstract) and by one of the 17 disease categories (searched, with all their descendant in the MeSH tree, among the MeSH terms describing the article).
- once this set of articles is identified, it is possible to exploit the already performed literature analysis in order to weight the importance of each candidate gene in the context. In this case the context is represented by the main disease associated to a particular family of other diseases that represent the comorbidities of the real patients.

The result of the first analysis (i.e. the patient-specific one) is an augmented ranked list of genes related to the patient case, where not all the suggested genes might belong to the list of the ones previously known as associated to the specific disease (DCM in our case). This could, therefore, lead to the discovery of new gene/disease associations.

The second type of analysis, instead, provides a general overview of the relationship between the main disease and the macro-categories of possible comorbidities; the result of this process leads to 17 prioritization lists (one for each comorbidity subgroup) of the candidate genes, where each gene is scored with the percentage of articles citing it on the total number of articles belonging to the set.

Both prioritization strategies exploit the LM-DB that actually contains more than 1,100,000 analyzed articles; in particular, for the scope of this analysis, we considered the extracted UMLS concepts belonging to the semantic class "Genes & Molecular Sequences" and the Gene annotations.

We will show two patient-specific gene prioritization examples, in order to point out both potentialities and current limits of this kind of analysis. The first phenotype, called Case 1, presents, besides DCM, "Creatine Phosphokinase Serum Increased" (sCPK Incr.), while the second one, Case 2, has DCM and "Left Ventricular Non-Compaction" (LVNC). The cases' description is reported in Table 6.3.

The data relative to these cases were used to generate a query to PubMed; the query generation process took advantage, for each feature, of the synonyms present in UMLS.

Table 6.3: Description of the two phenotypes used for testing the literature based gene prioritization. The percentages associated to the mutations are related to our real patient set and represent the frequency of each mutation among the real cases that are defined by the phenotypes.

	Case 1	**Case 2**
Mutation	Dystrophin (DMD Gene) 50% Lamin (LMNA Gene) 29% Other genes 21%	Lamin (LMNA Gene) 7% Tafazzin (TAZ GEne) 3% Other genes 91%
Features (UMLS codes)	Cardiomyopathy, Dilated (C0007193) Creatine Phosphokinase Serum Increased (C0241005)	Cardiomyopathy, Dilated (C0007193) Left Ventricular Non-Compaction (C1960469)

The resulting article sets were made up, respectively, by 30 articles for Case 1 and 40 articles for Case 2; for each article we considered the genes cited in its abstract and then we built a weighted list of genes for the two cases in exam. The obtained gene lists are represented in Table 6.4.

Table 6.4: List of candidate genes for the two phenotypes obtained by the literature analysis. The actual mutated genes are reported in bold.

Case 1		**Case 2**	
Gene	**# Occurrences**	**Gene**	**# Occurrences**
DMD	9	**TAZ**	2
LMNA	3	MYH7	2
DES	1	ACTC1	1
FKRP	1	**LMNA**	1
LAMA2	1	TNNT2	1

For Case 1, the results show that the *sCPK Incr.* feature performs well as a predictor of the possible mutations observed in our patient set; in fact, the frequency of cases with LMNA or DMD gene mutation (that are the most cited genes in the related literature), among the patients with DCM and sCPK Increased, is 79%. This good performance is due, besides the very nature of the considered feature as a mutation predictor, also to current

status of the scientific literature, where the association between DCM, *sCPK Incr.* and these mutations is strong.

The results obtained with Case 2, instead, show the actual limitations of this approach. In fact only 10% of the real patients' mutations (TAZ and LMNA gene) were identified; the reasons of this performance are mainly due to the inadequacy of the features to be a good mutation predictor and also to the existing scientific literature, where most of the mutated genes of our patients' set (91%) have not yet associated with DCM and LVNC.

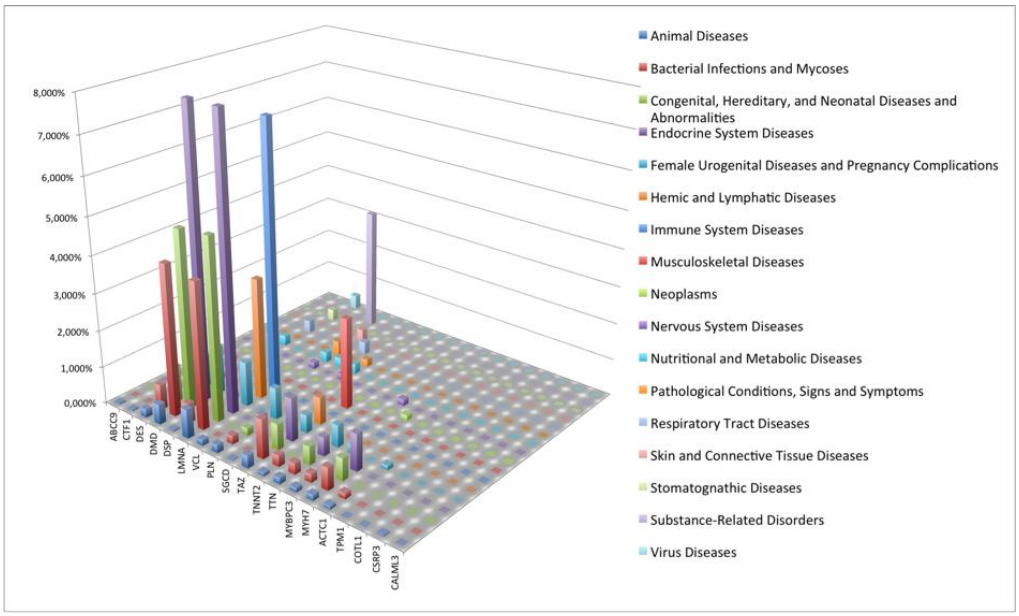

Figure 6.2: Graphical representation of the gene/comorbidity association. This can be used to derive the gene prioritization lists on the basis of the co-citation frequency of genes and comorbidity categories.

In order to test the second approach, we choose, among the 26 available, the 17 disease categories which could better cover the spectrum of the different types of comorbidities that can occur with DCM. These categories (Table 6.5) were used along with DCM to compose the queries to be sent to PubMed. The 17 prioritization lists so far obtained can be graphically represented as bar charts showing the gene/comorbidity association frequencies (Figure 6.2).

The results show that, for different phenotypes (DCM + comorbidities), it is possible to identify different prioritization lists among the candidate genes selected by the physicians. This clearly shows that, as soon as comorbidities are observed on the single patient, it is possible to update/rank the gene list.

Table 6.5: List of 17 disease categories used to prioritize genes to be tested for a mutation.

Categories of Comorbidities	
Pathological Conditions, Signs and Symptoms	Immune System Diseases
Congenital, Hereditary, and Neonatal Diseases and Abnormalities	Endocrine System Diseases
Nervous System Diseases	Respiratory Tract Diseases
Musculoskeletal Diseases	Neoplasms
Animal Diseases	Female Urogenital Diseases and Pregnancy Complications
Nutritional and Metabolic Diseases	Bacterial Infections and Mycoses
Skin and Connective Tissue Diseases	Stomatognathic Diseases
Hemic and Lymphatic Diseases	Substance-Related Disorders
Virus Diseases	

6.1.4. Literature Based Discovery

We chose to validate the LBD system (Section 5.2.6) on already known relations because the validation process of such systems is complex and would need the achieved results to be further confirmed with biological experiments; in particular, we applied our approach in the context of DCM where the goal is to discover new gene-disease associations. At present time, literature reports a list of several genes mutations currently recognized as responsible of DCM [179], so our validation process consisted in trying to discover for each gene its association to DCM from the scientific literature prior to its first publication. The test has been done on a subset of 17 genes and for each of them we followed several steps.

Publication date filter
To limit the analysis to the literature available before the gene-disease association it is necessary to correctly identify its first publication date. To this aim we build a PubMed query that refers not only to the complete gene name, but also to the most diffused synonyms of the gene, in order to be sure to have identified the first appearance of the association (i.e. co-citation) of the gene and DCM.

A→B step

Given the literature on DCM (concept A) available before the publication of the first gene-DCM association, the goal of the first step of the discovery process is to identify the concepts B related to DCM. In order to reduce the number of documents considered, the literature relative to DCM comprises only articles indexed with the MeSH term "Cardiovascular Disease" or those terms that are hierarchically dependent from it. From the overall set of B concepts cited in this literature, we remove those which are over-cited in order to exclude the concepts that are not enough specific and which informative content is therefore poor; this operation consisted in removing the concepts that are cited more than a specific number of times, empirically identified in 100.000. Afterwards we take into account exclusively those B concepts that belong to the list of Semantic Types reported in Table 6.6; we have defined this list with the goal of including all the concept types that could be most likely the intermediate concept between DCM and a gene and, on the other side, to exclude those concepts that, despite being part of UMLS, are too distant from this scope.

Table 6.6: Semantic types of B concepts (TUI is the Type Unique Identifier)

TUI	Semantic Type
T028	Gene or Genome
T114	Nucleic Acid, Nucleoside, or Nucleotide
T019	Congenital Abnormality
T116	Amino Acid, Peptide, or Protein
T048	Mental or Behavioral Dysfunction
T047	Disease or Syndrome
T033	Finding
T046	Pathologic Function
T085	Molecular Sequence
T043	Cell Function
T086	Nucleotide Sequence
T191	Neoplastic Process
T190	Anatomical Abnormality
T038	Biologic Function
T049	Cell or Molecular Dysfunction
T034	Laboratory or Test Result
T184	Sign or Symptom
T020	Acquired Abnormality
T087	Amino Acid Sequence
T200	Clinical Drug
T088	Carbohydrate Sequence

Furthermore, support and confidence of each A→B association have been evaluated; since we had not at our disposal any reference value for these indexes, we chose to set as a threshold, for both support and confidence, the average value of these indexes and to exclude from the next phase those concepts that were under these thresholds for at least one of the scores.

B → C step

Starting from the selected B concepts, keeping the same filter on publication dates used in the A→B step, we identify the whole literature related to this set and then the system extracts C concepts; in order to identify genes, this time only concepts belonging to the "Gene or Genome" Semantic Type have been used. At the end of these steps the system returns the list of the genes that, despite never being co-cited with DCM in the considered time span, could be connected with it. Each element of this list is characterized by the three described indexes: support, confidence relative to the B→C association and the heuristic score relative to the complete A→C association. In particular, for the sake of evaluating the potential new knowledge, we sorted the C concepts on the basis of their heuristic score. The results are shown in Table 6.7 and Table 6.8.

Table 6.7: Time spans valid for the discovery and number of papers and concepts found.

Gene	First date	First date with DMC	Related concepts	# Paper
TNNT2	15/05/94	2000 Jan	Not Found	5
TTN	01/08/75	1994 Oct	64	546
MYBPC3	15/02/93	1997 Mar	Not Found	17
ACTC	25/10/75	1998 May	98	1313
TPM1	07/06/05	2000 Jan	Not Found	51
MYH7	17/02/89	2000 Jan	Not Found	35
ABCC9	03/10/00	2004 Apr	Not Found	9
CLP	13/09/91	1997 Feb	Not Found	11
DES	12/12/76	1990 Jan	82	943
DMD	04/05/78	1990 Feb	35	290
DSP	04/06/05	2000 Oct	189	313
LDB3	13/02/98	2003 Dec	Not Found	14
LMNA	25/05/82	1999 Dec	166	214
MVCL	07/06/05	1997 Jan	Not Found	30
PLN	28/05/05	1990 May	45	203
SGCD	20/08/99	1999 Aug	Not Available	2
TAZ	11/07/91	1996 Apr	Not Found	8

Table 6.8: Summary results of the system validation for the genes the system is able to discover as associated to DCM.

Gene	Score	Support	Rank Supp	Rank Score
TTN	26832	92	68/542	41/542
ACTC	203577	1025	7/662	6/662
DES	21598	150	11/349	8/349
DMD	15268	300	2/349	21/349
DSP	256598	1115	5/887	8/887
LMNA	252739	752	9/822	5/822
PLN	7906	47	69/380	75/380

The results obtained in the validation process confirmed that the algorithm implemented, which uses association rules and several (semantic and statistical) filters, is effective both in selecting the most relevant concepts and in removing less interesting ones, very critical aspects that could strongly affect the research results. The validation of the system demonstrates its efficacy, as it is able to replicate many known connections between genes and DCM. Moreover, the results show that the heuristic function implemented (score) is a valid measure of the concept relevance, better than other types (e.g. support), since it allows to verify which associations between the starting (A) and final concepts (C) are stronger on the basis of the intermediate concepts (B).

The validation results show also that the system cannot discover new connections if the time span between the first appearance of the concept in literature and the discovery is too short and the number of articles relative to the concept is small.

6.2. The ONCO-i2b2 Project

The CBR system has been used also within the ONCO-i2b2 project, an information technology initiative started by the University of Pavia (Italy) and the IRCCS Fondazione Salvatore Maugeri (FSM) hospital in Pavia to support clinical research in oncology; this project aims at supporting translational research in oncology and exploits the software solutions implemented by the Informatics for Integrating Biology and the Bedside (i2b2) research center.

Within this project the ONCO-i2b2 data warehouse query tools have been extended with the distance score algorithm in order to enhance the patient selection process. The achieved results are shown, not in order to evaluate the CBR system itself, but to show how these systems can be integrated with a data warehouse that is currently used in the clinical and research practice.

The ONCO-i2b2 project is described in detail in [187] and [188]. The following sections are organized as follows: 6.2.1 introduces i2b2, 6.2.2 introduces the ONCO-i2b2 project along with its clinical issues and its IT infrastructure, finally 6.2.3 describes the integration between the ONCO-i2b2 data warehouse and the CBR system.

6.2.1. i2b2

The objectives for the NIH include the seamless integration of data obtained from clinical research, patient care and medical knowledge in order to build up a cornerstone for future research activities. The integration of data from health care and clinical research may facilitate the recruitment of subjects in clinical studies, evidence-based medicine and population monitoring.

The Informatics for Integrating Biology and the Bedside project (i2b2) [39] is one of the initiatives under the patronage of the NIH Roadmap National Centers for Biomedical Computing. i2b2, started in 2004, has set the goal of providing clinical researchers with the necessary software for collection and management of data coming from clinical research projects in the era of genomics. The final aim of the project is to develop a scalable information technology structure, which allows the use of data from clinical practice and their re-use for scientific research purposes.

Information coming from clinical research pass through three phases: extraction, validation and organization in a data warehouse. After the extraction from medical records, data must be validated. The extraction phase can be applied on many data sources; for example, if researchers want to know whether a patient is a smoker, this information can be directly read from different clinical documents or extracted with natural language processing algorithms. Afterwards, the information are organized in a data warehouse and, finally, displayed in order to enable their analysis. From a software perspective, the developer team has created the i2b2 Hive, a framework where several web services can be linked together to create specific workflows. The i2b2 software is characterized by a client-server architecture with the Hive on the server side and an application called Workbench installed or accessible from the browser, on the client side. The integration strategy of i2b2 applications is based on the Service-Oriented Architecture (SOA), which allows the different software modules to communicate between each other with XML messages. The Hive can be considered as a set of Web Services aggregated to provide a number of functionalities onto data coming from the i2b2 data warehouse, called Clinical Research Chart (CRC). Each Web Service defines a functional component, which can be used independently form the others. Different Web Services integrated in the Hive are called "Cells”. The CRC is a conceptual model to systematically structure data of clinical research in a coherent collection that can be used for research purposes. To provide this model, the i2b2 Hive was developed with the dual purpose of collecting

data from heterogeneous and complex sources and, at the same time, to keep a simple interface for researchers.

6.2.2. The Project

Most diseases, including cancer, involve a large number and a variety of elements that interact via complex networks. Therefore, simply observing the problem from a clinical or biological point of view is not sufficient for the treatment of such diseases. Moreover, effective treatment of disease requires the clinicians to consider the effects of a patient's personal genetic background. Personalized medicine is a proposed approach to develop treatment regimens that take into account each patient's unique genetic profile, allowing the treatment to fit the specific needs of patient sets with different genetic backgrounds. Gathering and integrating data coming from both clinical practice and research settings, together with the possibility of expanding the patients cohort of interest using an innovative case-based reasoning approach, offer physicians the possibility to view the problem in a more complete way. The ONCO-i2b2 project is aimed at developing a bioinformatics platform designed to integrate clinical and research data to support translational research in oncology.

The ONCO-i2b2 system gathers data from the FSM pathology unit (PU) database and from the hospital biobank, and integrates them with clinical information from the hospital information system (HIS) [189].

Complexity is just one of the problems faced in the integration process leaded in order to provide a robust integrated research environment. The system imports data collected during the clinical practice. These data are often not properly structured and may require further extraction steps (e.g. natural language processing tools may be applied to medical reports in order to extract information from clinical narratives). In the database are also integrated -omics data resulting from high-throughput measurement technologies. Although well formatted, such data pose also an integration challenge because of their sheer volume.

Addressing these different challenges of complexity, scope and scale requires a dedicated integration architecture.

6.2.3. Case Based Reasoning

The CBR system (Section 5.3) has been used in the ONCO-i2b2 project in order to allow researchers to enhance the patient selection process with an information retrieval procedure that uses the whole medical concept space related to a patient set to identify a group of similar patients. This functionality supports the extension of the original patient set obtained with the i2b2 query tool, to patients that prove to have a high degree of closeness.

The i2b2 concepts that characterize the cases stored into the ONCO-i2b2 data warehouse have been mapped into the CBR system internal case representation based on the UMLS Metathesaurus in order to make them usable by the CBR system. After a patient set has been retrieved using the i2b2 query tool, our procedure finds all concepts related to patients' observations by means of an array containing UMLS concepts, described by their CUI and the boolean modifier (see Section 5.3.1). The distance scores between each patient and the others of this set are then calculated to retrieve similar patients and obtain additional information that can support new insights in the study of tumors.

The example showed in Figure 6.3 represents the results of the method described above. Thanks to a dedicated plug-in, developed for the i2b2 web client interface using the JavaScript InfoVis Toolkit [190], the user is able to:

1. select a pre-defined patient set that represents the cases related to the medical problem of interest;
2. find the most similar cases using the ONCO-i2b2 data warehouse (a threshold can be set in order to filter the search);
3. visualize the results on a graph that represents the distance of each new patient to the selected patient set. Patients of the original patient set with the same distance are clustered and represented in a network.

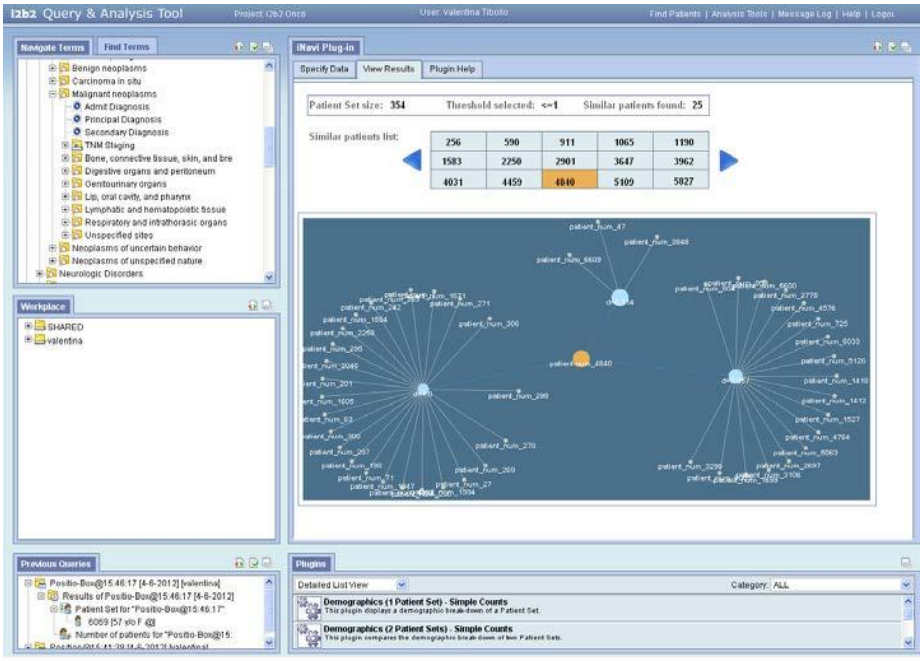

Figure 6.3: I2b2 web client with the novel plug-in developed for finding similar patients given a specific patient set. The plug-in shows how the selected new patient (orange cell in the table at the top) is similar to the patients belonging to the patient set. The most similar patients belong to a

group of three patients that have a distance d=0.714. The other two groups have d=0.857 (group on the right side) and d=1.0 (group on the left side)

Using this plug-in, a researcher can dynamically find similar patients and explore their related concepts in order to collect useful information potentially related to the medical problem of interest.

Chapter 7

Conclusions and Future Works

The aim of the work described in this thesis was to investigate which techniques allow to manage the information coming from several areas of scientific research and to explore their adaptability to the emerging translational scenario. In particular, the focus was set on the management, analysis and exploitation of the data coming from basic and patient-oriented research.

The main characteristic of the considered information is its huge volume; in fact, data coming both from basic research, such as the scientific literature, and patient-oriented research, such as patients' data generated by healthcare organizations, are characterized by an ever increasing size. Many new articles are published on scientific journals every day and the overall publication rate has been proved to be regularly rising; on the other side, also a lot of patients' data are collected daily and, since the investigation methodologies become more and more sophisticated and the accessibility to new types of data, such as *–omics* data, becomes easier and easier, also their increasing volume is an important issue to deal with.

Another characteristic of the considered data is their heterogeneity; in fact, scientific literature contains many types of entities relevant for biomedical research (e.g. genes, proteins, drugs, diseases), while every healthcare organization typically manages its own patients' data autonomously according to the codification systems that better suite its specific purposes.

The management of these two types of data has been dealt with techniques with strong traits of automation and flexibility; automatic analysis techniques relieve the user of facing directly the huge volume of data, while flexibility is needed in order to deal with their heterogeneity. In particular, Information Extraction techniques have been used to automatically analyze the abstracts of MEDLINE articles, accessed through

PubMed, in order to extract many types of cited entities, while Case Based Reasoning techniques have been employed to assess a score to measure the relative distance between patient cases on the basis of their heterogeneous features in order to identify, given a new patient, those cases that prove to be similar and which information can be useful to solve it.

The implemented Literature Mining system performs several tasks. First, it downloads the relevant scientific literature from PubMed, exploiting the ENTREZ web services provided by the NCBI; the retrieved articles are stored in the Literature Mining Database (LM-DB), an *ad hoc* database designed to provide a reliable persistence layer to the whole system. Once the articles are available, their abstracts are analyzed by the core of the Literature Mining system, that employs Text Mining techniques to extract relevant biomedical entities, such as genes, proteins and diseases, which are then stored, along with the basic articles' data, in the LM-DB. The Information Extraction process is based both on a standard lexical and syntactic analysis of the text and on the use of specific biomedical terminologies to match the analyzed text with the concepts to extract. Exploiting the LM-DB adequately populated with articles and cited entities, the Literature Mining system can be used for different purposes: a) the summarization of a given set of articles through the set of relevant entities contained in their abstracts, b) the retrieval of scientific articles according to the concepts cited into their abstracts and c) the discovery of potential new knowledge through the evaluation of indirect associations reported in literature. The latter task is performed with Literature Based Discovery techniques that exploit the literature analysis and its results stored in the LM-DB.

The first goal of the Case Based Reasoning system is to assess a "common ground" on which to compare different medical cases; this operation, called *mapping*, exploits the Unified Medical Language System (UMLS) to have a common codification system available to represent all the disparate features of a patient with the necessary consistency. Having the cases mapped onto this common layer, the system can apply the developed distance score algorithm to evaluate the relative degree of closeness between two patients; the algorithm exploits the internal case representation of the system and produces a global distance score on the basis of distances between the features of the compared cases; to calculate the distance between two features the system evaluates the length of the shortest paths that connects them in UMLS Metathesaurus. Once it is possible to assess the relative distance between two cases, the Case Based Reasoning system can select, given a new patient, the already-known cases similar to him in order to exploit their information to solve the incoming case; in alternative, the distance algorithm can be used to produce the distance matrix for a given set of cases and evaluate their clustering on the basis of the set of their features.

Along with the two main systems, also a Graphical User Interface (GUI) has been developed in order to provide an intuitive access to some of their

functionalities, such as the navigation of the literature enhanced by the use of biomedical terminologies and the assessment of the distance scores between a set of cases. Moreover, the GUI prepares the ground for the actual integration of the two systems. Currently the GUI generates a PubMed query on the basis of the features of a real patient case, thus allowing a patient-oriented literature interrogation.

The evaluation of the adopted techniques and the implemented systems has been performed within two research projects with strong translational traits: the INHERITANCE project and the ONCO-i2b2 project. INHERITANCE (Integrated Heart Research In Translational Genetics of Cardiomyopathies in Europe) is a multidisciplinary, multi-center research project funded by the European Commission that seeks to study the genetics of inherited Dilated Cardiomyopathy (DCM) and to understand the impact and management of the condition within families that suffer from DCMs. The project is structured into six research areas that study different facets of the DCM condition and its translational strategy is based on a clinical algorithm that seeks to determine disease-specific features to be associated with different types of DCM or suggest specific genetic or metabolic pathways of disease.

ONCO-i2b2 is an information technology initiative started by the University of Pavia (Italy) and the IRCCS Fondazione Salvatore Maugeri hospital in Pavia to support clinical research in oncology; this project aims at supporting translational research in oncology and exploits the software solutions implemented by the Informatics for Integrating Biology and the Bedside (i2b2) research center in the U.S.

The Literature Mining system has been evaluated within INHERITANCE; it has been used to extend the set of candidate DCM-causing genes with new, still-not-evaluated, genes on the basis of the entities extracted from each gene-specific literature. Moreover, the performed literature analysis has been exploited to prioritize the set of DCM-causing genes starting from a single case or, in alternative, with the goal of achieving a general prioritization list for the disease. Finally, the Literature Based Discovery system has been tested within this project, by demonstrating that it is able to infer most of the associations between DCM and its causative genes by exploiting only the literature prior to their first appearance.

The Case Based Reasoning system has been tested on both projects with different objectives: in INHERITANCE with the aim of grouping a set of simulated DCM patients on the basis of their features. This test has shown the CBR system capability to cluster patients with similar diagnoses on the basis of their closeness. In ONCO-i2b2 the purpose was to enhance the project's data warehouse's query tool in order to extend the patient selection process on the basis of the semantic distance score. The actual aim of this evaluation was to show how the Case Based Reasoning system can be integrated within a data warehouse query system currently used in the clinical and research practice.

The performed evaluation showed promising results; in particular, it has become clear how the systems flexibility in properly managing the data, affects the many different environments in which the developed techniques are applicable. In fact, the systems have been used to perform several tasks and proved a high degree of adaptability; moreover, the techniques adopted and their actual implementation grant also a high degree of extensibility. For instance the Literature Mining system can be integrated with new Information Extraction pipelines in order to extract new types of concepts from the scientific literature, while the very nature of the Case Based Reasoning system, with its case representation based on UMLS, allows new cases from new sources to be added to the overall reasoning environment.

Future directions of the research described in this thesis comprehend: the extension and refinement of the Text Mining techniques employed in the Literature Mining system to allow the extraction of more complex data patterns, the evaluation of new distance algorithms in order to allow the system to consider different types of features (e.g. clinical measures and genomic data). Moreover, the evaluation of the CBR system must be carried on, in particular to test its performance when comparing patients characterized by a number of features that largely overcomes the ones that it has been possible to test until now. Alongside this methodological effort, it is necessary to continue the implementation of the system Graphical User Interface in order to properly integrate all the functionalities of the two systems and to complete a tool that can be actually used in the clinical and research practice.

References

[1] Bush V: “Science The Endless Frontier: A Report to the President Government Printing Office”. 1945, 3(3). - http://www.nsf.gov/about/history/nsf50/vbush1945_content.jsp (accessed January 9, 2013).

[2] Glossary of Terms for Human Subjects Protection and Inclusion Issues, based on the 1997 Report of the NIH Director’s Panel on Clinical Research, entry: “clinical research”. - http://grants.nih.gov/grants/peer/tree_glossary.pdf (accessed January 9, 2013).

[3] Rubio DM, Schoenbaum EE, Lee LS, Schteingart DE, Marantz PR, Anderson KE, Platt LD, Baez A, Esposito K: “Defining Translational Research: Implications for Training”. Academic Medicine 2010.

[4] National Institutes of Health: “Definitions under Subsection 1 (Research Objectives), Section I (Funding Opportunity Description), Part II (Full Text of Announcement), of RFA-RM-07-007: Institutional Clinical and Translational Science Award (U54)”. 2007. - http://grants.nih.gov/grants/guide/rfa-files/RFA-RM-07-007.html (accessed January 9, 2013).

[5] Sarkar IN: “Biomedical informatics and translational medicine”. J Transl Med 2010, 8:22.

[6] Osheroff JA, Teich JM, Middleton B, Steen EB, Wright A, Detmer DE: “A roadmap for national action on clinical decision support”. J Am Med Inform Assoc 2007, 14:141-145.

[7] Kundaje A, Middendorf M, Shah M, Wiggins CH, Freund Y, Leslie C: “A classification-based framework for predicting and analyzing gene regulatory response”. BMC Bioinformatics 2006, 7(Suppl 1):S5.

[8] Downing GJ, Boyle SN, Brinner KM, Osheroff JA: "Information management to enable personalized medicine: stakeholder roles in building clinical decision support". BMC Med Inform Decis Mak 2009, 9:44.

[9] Kawamoto K, Lobach DF, Willard HF, Ginsburg GS: "A national clinical decision support infrastructure to enable the widespread and consistent practice of genomic and personalized medicine". BMC Med Inform Decis Mak 2009, 9:17.

[10] Berlin A, Sorani M, Sim I: "A taxonomic description of computer-based clinical decision support systems". J Biomed Inform 2006, 39:656-667.

[11] Wright A, Bates DW, Middleton B, Hongsermeier T, Kashyap V, Thomas SM, Sittig DF: "Creating and sharing clinical decision support content with Web 2.0: Issues and examples". J Biomed Inform 2009, 42:334-346.

[12] MEDLINE/PubMed Baseline Repository – http://mbr.nlm.nih.gov/. (accessed January 9, 2013).

[13] Bodenreider O: "Biomedical ontologies in action: role in knowledge management, data integration and decision support". Yearb Med Inform 2008, 67-79.

[14] Hammond WE: "Health Level 7: an application standard for electronic medical data exchange". Top Health Rec Manage 1991, 11:59-66.

[15] Wade G, Rosenbloom ST: "The impact of SNOMED CT revisions on a mapped interface terminology: terminology development and implementation issues". J Biomed Inform 2009, 42:490-493.

[16] WHO International Classification of Diseases – http://www.who.int/classifications/icd/en/. (accessed January 9, 2013).

[17] Prokosch HU, Ganslandt T: "Perspectives for medical informatics. Reusing the electronic medical record for clinical research". Methods Inf Med 2009, 48:38-44.

[18] Lindberg DA, Humphreys BL, McCray AT: "The Unified Medical Language System". Methods Inf Med 1993, 32:281-291.

[19] PubMed - http://www.pubmed.gov. (accessed January 9, 2013).

[20] Galperin MY, Cochrane GR: "Nucleic Acids Research annual Database Issue and the NAR online Molecular Biology Database Collection in 2009". Nucleic Acids Res 2009, 37:D1-D4.

[21] Maviglia SM, Yoon CS, Bates DW, Kuperman G: "KnowledgeLink: impact of context-sensitive information retrieval on clinicians' information needs". J Am Med Inform Assoc 2006, 13:67-73.

[22] Jacquemart P, Zweigenbaum P: "Towards a medical question-answering system: a feasibility study". Stud Health Technol Inform 2003, 95:463-468.

[23] Ruttenberg A, Clark T, Bug W, Samwald M, Bodenreider O, Chen H, Doherty D, Forsberg K, Gao Y, Kashyap V, Kinoshita J, Luciano J, Marshall MS, Ogbuji C, Rees J, Stephens S, Wong GT, Wu E, Zaccagnini D, Hongsermeier T, Neumann E, Herman I, Cheung KH: "Advancing translational research with the Semantic Web". BMC Bioinformatics 2007, 8(Suppl 3):S2.

[24] Mangalampalli A, Rama C, Muthiyalian R, Jain AK: "High-end clinical domain information systems for effective healthcare delivery". Int J Electron Healthc 2007, 3:208-219.

[25] Sax U, Schmidt S: "Integration of genomic data in Electronic Health Records–opportunities and dilemmas". Methods Inf Med 2005, 44:546-550.

[26] Hoffman MA: "The genome-enabled electronic medical record". J Biomed Inform 2007, 40:44-46.

[27] Altman RB: "Translational bioinformatics: linking the molecular world to the clinical world". Clin Pharmacol Ther 2012, 91:994–1000.

[28] Ashley EA, Butte AJ, Wheeler MT, Chen R, Klein TE, Dewey FE, Dudley JT, Ormond KE, Pavlovic A, Morgan AA, Pushkarev D, Neff NF, Hudgins L, Gong L, Hodges LM, Berlin DS, Thorn CF, Sangkuhl K, Hebert JM, Woon M, Sagreiya H, Whaley R, Knowles JW, Chou MF, Thakuria JV, Rosenbaum AM, Zaranek AW, Church GM, Greely HT, Quake SR, Altman RB: "Clinical assessment incorporating a personal genome". Lancet 2010, 375:1525–1535.

[29] Ritchie MD, Denny JC, Crawford DC, Ramirez AH, Weiner JB, Pulley JM, Basford MA, Brown-Gentry K, Balser JR, Masys DR, Haines JL, Roden DM: "Robust replication of genotype-phenotype associations across multiple diseases in an electronic medical record". Am J Hum Genet 2010, 86(4):560-572.

[30] Eriksson N, Macpherson JM, Tung JY, Hon LS, Naughton B, Saxonov S, Avey L, Wojcicki A, Pe'er I, Mountain J: "Web-based, participant-driven studies yield novel genetic associations for common traits". PLoS Genet 2010, 6(6): e1000993.

[31] The Wellcome Trust Consortium: "Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls". Nature 2007, 447:661–678.

[32] Ng SB, Buckingham KJ, Lee C, Bigham AW, Tabor HK, Dent KM, Huff CD, Shannon PT, Jabs EW, Nickerson DA, Shendure J, Bamshad MJ: "Exome sequencing identifies the cause of a mendelian disorder". Nat Genet 2010, 42:30–35.

[33] McGary KL, Park TJ, Woods JO, Cha HJ, Wallingford JB, Marcotte EM: "Systematic discovery of nonobvious human disease models through orthologous phenotypes". Proc Natl Acad Sci USA 2010, 107:6544–6549.

[34] Kuhn M,Campillos M, Letunic I, Jensen LJ, Bork PA: "Side effect resource to capture phenotypic effects of drugs". Mol Syst Biol 2010, 6:343.

[35] Shah NH, Jonquet C, Chiang AP, Butte AJ, Chen R, Musen MA: "Ontology-driven indexing of public datasets for translational bioinformatics". BMC Bioinformatics 2010, 10(2):S1.

[36] Duarte NC, Becker SA, Jamshidi N, Thiele I, Mo ML, Vo TD, Srivas R, Palsson BØ: "Global reconstruction of the human metabolic network based on genomic and bibliomic data". Proc Natl Acad Sci USA 2007, 104:1777-1782.

[37] Goh KI, Cusick ME, Valle D, Childs B, Vidal M, Barabási AL: "The human disease network". Proc Natl Acad Sci USA 2007, 104: 8685-8690.

[38] Xu H, Schaniel C, Lemischka IR, Ma'ayan A: "Toward a complete in silico, multi-layered embryonic stem cell regulatory network". Wiley Interdiscip Rev Syst Biol Med 2010, 2:708-733.

[39] Murphy SN, Weber G, Mendis M, Gainer V, Chueh HC, Churchill S, Kohane I: "Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2)". J Am Med Inform Assoc 2010, 17:124-130.

[40] Dudley JT, Pouliot Y, Chen R, Morgan AA, Butte AJ: "Translational bioinformatics in the cloud: an affordable alternative". Genome Med 2010, 2:51.

[41] Homer N, Szelinger S, Redman M, Duggan D, Tembe W, Muehling J, Pearson JV, Stephan DA, Nelson SF, Craig DW: "Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays". PLoS Genet 2008, 4:e1000167.

[42] Nyholt DR, Yu CE, Visscher PM: "On Jim Watson's APOE status: genetic information is hard to hide". Eur J Hum Genet 2009, 17:147–149.

[43] Pulley J, Clayton E, Bernard GR, Roden DM, Masys DR: "Principles of human subjects protections applied in an opt-out, de-identified biobank". Clin Transl Sci 2010, 3:42–48.

[44] Ananiadou S, Mc Naught J: "Introduction". In "Text Mining for Biology and Biomedicine". Artech House Publishers 2006, 1:1-12.

[45] Hahn U, Mc Wermter J: "Levels of Natural Language Processing for Text Mining". In "Text Mining for Biology and Biomedicine". Artech House Publishers 2006, 2:13-41.

[46] Jurafsky D, Martin JH: "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition". Prentice-Hall 2000.

[47] Arens R: "A Preliminary Look into the Use of Named Entity Information for Bioscience Text Tokenization". Proc HLT-NAACL Student Research Workshop 2004, 37–42.

[48] Park JC, Kim J: "Named Entity Recognition". In "Text Mining for Biology and Biomedicine". Artech House Publishers 2006, 6:121-142.

[49] Porter MF: "An Algorithm for Suffix Stripping". Program 1980, 14(3):130–137.

[50] Brill E: "Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging". Computational Linguistics 1995, 21(4):543–565.

[51] Brants T: "TnT: A Statistical Part-of-Speech Tagger". Proc 6th Conf on Applied Natural Language Processing 2000, 224–231.

[52] Ratnaparkhi A: "A Maximum Entropy Part-of-Speech Tagger". Proc 1st Conf on Empirical Methods in Natural Language Processing 1996, 133–141.

[53] Giménez J, Márquez L: "Fast and Accurate Part-of-Speech Tagging". Proc Int Conf on Recent Advances in Natural Language Processing 2003, 158-165.

[54] Hahn U, Mc Wermter J: "High-Performance Tagging on Medical Texts". Proc 20th Intl Conf on Computational Linguistics 2004, 973-979.

[55] Ramshaw LA, Mitchell PM: "Text chunking using transformation-based learning". In ACL Third Workshop on Very Large Corpora 1995, 82–94.

[56] Molina A, Pla F: "Shallow Parsing Using Specialized HMMs". Journal of Machine Learning Research 2002, 2(4):595–613.

[57] Kudo T, Matsumoto T: "Chunking with Support Vector Machines". Proc 2nd Meeting of the North American Chapter of the Association for Computational Linguistics 2001, 192–199.

[58] Kang N, van Mulligen EM, Kors JA: "Comparing and combining chunkers of biomedical text". Journal of Biomedical Informatics 2011, 44(2):354–360.

[59] Shatkay H, Feldman R: "Mining the Biomedical Literature in the Genomic Era: An Overview". J Comput Biol 2003, 10(6):821–855.

[60] Bodenreider O: "Lexical, Terminological, ans Ontological Resources for Biological Text Mining". In "Text Mining for Biology and Biomedicine". Artech House Publishers 2006, 3:43-66.

[61] Corney DP, Buxton BF, Langdon WB, Jones DT: "BioRAT: Extracting Biological Information from Full-Length Papers". Bioinformatics 2004, 20(17):3206–3213.

[62] Hahn U, Romacker M, Schulz S: "Medsyndikate - A Natural Language System for the Extraction of Medical Information from Findings Reports". Int J Med Inform 2002, 67(1–3):63–74.

[63] Medical Subject Headings (MESH®) Fact Sheet - http://www.nlm.nih.gov/pubs/factsheets/mesh.html (accessed January 10, 2013).

[64] UMLS® Reference, SPECIALIST Lexicon and Lexical Tools - http://www.ncbi.nlm.nih.gov/books/NBK9680 (accessed January 10, 2013).

[65] Fellbaum C: "WordNet: An Electronic Lexical Database, Language, Speech, and Communication". MIT Press 1998.

[66] Leroy G, Chen H: "Meeting Medical Terminology Needs-The Ontology-Enhanced Medical Concept Mapper". IEEE Trans on Inf Technol Biomed 2001, 5(4):261–270.

[67] The Gene Ontology Consortium: "Gene ontology: tool for the unification of biology". Nat Genet 2000, 25(1):25-29.

[68] Baader F, Calvanese D, McGuinness D, Nardi D, Patel-Schneider P: "The Description Logics Handbook". Cambridge University Press 2003.

[69] Franzén K, Eriksson G, Olsson F, Asker L, Lidén P, Cöster J: "Protein Names and How to Find Them" International Journal of Medical Informatics 2002, 67(1–3):49-61.

[70] "Obstacles of Nomenclature". Nature 1997. 389(6646):1.

[71] Zhou G, Zhang J, Su J, Shen D, Tan C: "Recognizing names in biomedical texts: a machine learning approach". Bioinformatics 2004, 20(7):1178-1190.

[72] Hirschman L, Morgan AA, Yeh AS: "Rutabaga by Any Other Name: Extracting Biomedical Name". Journal of Biomedical Informatics 2002, 35:247–259.

[73] Tuason O, Chen L, Liu H, Blake JA, Friedman C: "Biomedical Nomenclatures: A Source of Lexical Knowledge and Ambiguity". Proc. Pacific Symp. on Biocomputing 2004, 238-249.

[74] Tsuruoka Y, Tsujii J: "Improving the Performance of Dictionary-Based Approaches in Protein Name Recognition". Journal of Biomedical Informatics 2004, 37:461-470.

[75] Krauthammer M, Rzhetsky A, Morozov P, Friedman C: "Using BLAST for Identifying Gene and Protein Names in Journal Articles". Gene 2000, 259(1–2):245–252.

[76] Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ: "Gapped BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs". Nucleic Acids Research 1997, 25(17):3389–3402.

[77] Hanisch D, Fluck J, Mevissen HT, Zimmer R: "Playing Biology's Name Game: Identifying Protein Names in Scientific Text". Proc Pacific Symp on Biocomputing 2003, 403-414.

[78] Seal RL, Gordon SM, Lush MJ, Wright MW, Bruford EA: "genenames.org: the HGNC resources in 2011". Nucleic Acids Res 2011, 39:D519-D519.

[79] Online Mendelian Inheritance in Man, OMIM® - http://omim.org/ accessed January 10, 2013).

[80] The UniProt Consortium: "Reorganizing the protein space at the Universal Protein Resource (UniProt)". Nucleic Acids Res 2012, 40:D71-D75.

[81] Fukuda K, Tamura A, Tsunoda T, Takagi T: "Toward Information Extraction: Identifying Protein Names from Biomedical Papers". Proc Pacific Symp on Biocomputing 1998, 707–718.

[82] Gaizauskas R, Demetriou G, Humphreys K: "Term Recognition and Classification in Biological Science Journal Articles".. Proc Workshop on Computational Terminology for Medical and Biological Applications 2000, 37–44.

[83] Chomsky N; "Three models for the description of language". Information Theory, IEEE Transactions 1956, 2(3):113–124.

[84] Collier N, Nobata C, Tsujii J: "Extracting the Names of Genes and Gene Products with a Hidden Markov Model". Proc 17th Int Conf on Computational Linguistics 2000, 201-207.

[85] Tanabe L, Wilbur WJ: "Tagging Gene and Protein Names in Biomedical Text". Bioinformatics 2002, 18(8):1124-1132.

[86] Mika S, Rost B: "Protein Names Precisely Peeled off Free Text". Bioinformatics 2004, 20(1):i241–i247.

[87] Swanson DR: "Fish oil, Raynaud's syndrome, and undiscovered public knowledge". Perspectives in Biology and Medicine 1986, 30(1):7-18.

[88] Smalheiser NR, Swanson DR: "Using ARROWSMITH: a computer-assisted approach to formulating and assessing scientific hypotheses". Comput Methods Programs Biomed 1998, 57(3):149-53.

[89] Gordon MD, Lindsay RK: "Toward discovery support systems: A replication, reexamination, and extension of Swanson's work on literature-based discovery of a connec-tion between Raynaud's and fish oil". Journal of the American Society for Information Science 1996, 47(2):116-128.

[90] Weeber M, Klein H, de Jong-van den Berg LTW, Vos R: "Using concepts in literature-based discovery: Simulat-ing Swanson's Raynaud-fish oil and migraine-magnesium discoveries". Journal of the American Society for Information Science and Technology 2001, 52(7):548-557.

[91] Pratt W and Yetisgen-Yildiz M: "LitLinker: Capturing Connections across the Biomedical Literature". Proceedings of the International Conference on Knowledge 2003.

[92] Agrawal R, Imielinski T and Swami A: "Mining associations between sets of items in massive databases". Proceedings of the ACM-SIGMOD 1993.

[93] Srinivasan P, Libbus B: "Mining MEDLINE for implicit links between dietary substances and diseases". Bioinformatics 2004, 20(1):i290-i296.

[94] Hristovski D, Stare J, Peterlin B, Dzeroski S: "Supporting discovery in medicine by association rule mining in MEDLINE and UMLS". Medinformation 2001, 10(2):1344-1348.

[95] Schlank R: "Dynamic Memory: a Theory of Learning in Computers and People". Cambridge University Press 1982.

[96] Leake DB: "Case-based Reasoning: Experiences, Lessons and Future Directions". AAAI Press 1996.

[97] Aamodt A, Plaza E: "Case-based reasoning: foundational issues, methodological variations and system approaches". AI Communications 1994, 7(1):39-59.

[98] Lopez de Mantaras R: "Case-Based Reasoning". Machine Learning and Its Applications, Lecture Notes in Computer Science 2001, 2049:127-145.

[99] Dubois D, Prade H, Testemale C: "Weighted fuzzy pattern matching". Fuzzy Sets and Systems 1988, 28:351-362.

[100] Schweizer B, Sklar A: “Associative functions and abstract semi-groups”. Publicationes Mathematicae Debrecen 1963, 10:69-81.

[101] Cunningham P: “A taxonomy of similarity mechanisms for case-based reasoning”. Technical Report UCD-CSI-2008-01, University College Dublin 2008.

[102] Wiratunga N, Koychev I, Massie S: “Feature selection and generalisation for retrieval of textual cases”. ECCBR Lecture Notes in Computer Science 2004, 806–820.

[103] Gabrilovich E, Markovitch S: “Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge”. Proceedings of the 21st National Conference on Artificial Intelligence 2006, 1301–1306.

[104] Bergmann R, Stahl A: “Similarity measures for object-oriented case representations”. Lecture Notes in Computer Science 1998, 25–36.

[105] Sanders KE, Kettler BP, Hendler JA: “The case for graph-structured representations”. ICCBR Lecture Notes in Computer Science 1997, 1266:245–254.

[106] Minor M, Tartakovski A, Bergmann R: “Representation and structure-based similarity assessment for agile workflows”. ICCBR 2007, 4626:224 – 238.

[107] Shimazu H: “A textual case-based reasoning system using xml on the world-wide web”. Lecture Notes in Computer Science 1998, 1488:274–285.

[108] Stanfill C, Waltz DL: “Toward Memory-Based Reasoning” Comm ACM 1986, 29:1213-1228.

[109] Wilson D, Martinez T: “Improved Heterogeneous Distance Functions”. J Artificial Intelligence Research 1997, 6:1-34.

[110] Beygelzimer A, Kakade S, Langford J: “Cover Trees for Nearest Neighbor” Proc. 23rd Int’l Conf. Machine Learning 2006.

[111] Tanimoto T: “An Elementary Mathematical Theory of Classification and Prediction [Z]” technical report IBM Corp 1958.

[112] Jaccard P: “The Distribution of the Flora in the Alpine Zone” New Phytologist 1912, 11:37-50.

[113] Greene D, Tsymbal A, Bolshakova N, Cunningham P: "Ensemble Clustering in Medical Diagnostics," Proc. CBMS 2004, 576-581.

[114] Dice L, "Measures of the Amount of Ecologic Association between Species," Ecology 1945, 26:297-302.

[115] Wu Z, Palmer MS: "Verb Semantics and Lexical Selection," Proc Ann Meeting ACL 1994, 133-138.

[116] Needleman S, Wunsch C: "A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins," J Molecular Biology 1970, 48:443-453.

[117] Smith T, Waterman M: "Identification of Common Molecular Subsequences," J Molecular Biology 1981, 147:195-197.

[118] Bergmann R: "Experience Management: Foundations, Development Methodology, and Internet-Based Applications," Lecture Notes in Computer Science 2002, 2432.

[119] Delany SJ, Bridge DG: "Catching the Drift: Using Feature- Free Case-Based Reasoning for Spam Filtering". ICCBR 2007, 314-328.

[120] Li M, Badger JH, Chen X, Kwong S, Kearney PE, Zhang H: "An Information-Based Sequence Distance and Its Application to Whole Mitochondrial Genome Phylogeny" Bioinformatics 2001, 17:149-154.

[121] Resnik P: "Using Information Content to Evaluate Semantic Similarity in a Taxonomy" Proc IJCAI 1995, 448-453.

[122] Bichindaritz I, Marling C: "Case-based reasoning in the health sciences: what's next?". Artificial Intelligence in Medicine 2006, 36:127–135.

[123] Bichindaritz I: "Case based reasoning meets the semantic web in biology and medicine". Proceedings ECCBR 2004, 47-61.

[124] Vorobieva O, Schmidt R: "CBR investigation of therapy inefficacy". Proceedings of the 7th European conference on case-based reasoning 2004, 65-72.

[125] Jurisica I, Glasgow J: "Applications of case-based reasoning in molecular biology". AI 2004, 25(1):85-95.

[126] Wu D, Weber R, Abramson FC: "A case-based framework for leveraging NutriGenomics knowledge and personalized nutrition counseling". Proceedings of the 7th European conference on case-based reasoning 2004, 73-82.

[127] Nilsson M, Funk P, Sollenborn M: "Complex measurement classification in medical applications using a case-based approach". Proceedings of the 5th international conference on case-based reasoning 2003, 63-72.

[128] El Balaa Z, Strauss A, Uziel P, Maximini K, Traphoner R: "FM- Ultranet: a decision support system using case-based reasoning applied to ultrasonography". Proceedings of the 5th international conference on case-based reasoning 2003, 37-44.

[129] Montani S, Portinale L, Leonardi G, Bellazzi R: "Applying case- based retrieval to hemodialysis treatment". Proceedings of the 5th international conference on case-based reasoning 2003, 53-62.

[130] Bichindaritz I, Kansu E, Sullivan KM: "Case-based reasoning in CAREPARTNER: gathering evidence for evidence-based medical practice". Proceedings EWCBR 1998, 334-345.

[131] Bradburn C, Zeleznikow J: "The application of Cased-Based reasoning to the tasks of health care planning". Proceedings of European Workshop on CBR 1993, 365–378.

[132] Macura R, Macura K: "MacRad: radiology image resource with a Cased-Based retrieval system", Proceedings of 1st International Conference on CBR 1995, 43–54.

[133] Xu LD: "An integrated rule- and Cased-Based approach to AIDS initial assessment". International Journal of Biomedical Computing 1996, 40:197-207.

[134] Branting LK, Porter BW: "Rules and precedents as complementary warrants". Proceedings 9th AAAI 1991.

[135] Leake DB: "Combining rules and cases to learn case adaptation". Proceedings of 17th International Conference of Cognitive Science Society 1995.

[136] Gierl L, Stengel-Rutkowski S: "Integrating consultation and semi-automatic knowledge acquisition in a prototype-based architecture: experiences with Dysmorphic Syndromes", Artific Intell in Med 1994, 6:29–49.

[137] Nilsson M, Sollenborn M: “Advancements and trends in medical case-based reasoning: an overview of systems and system development”. Proceedings of the 17th international florida artificial intelligence research society conference—special track on case-based reasoning 2004, 178–183.

[138] Balaa ZE, Traphoner R: “Case-based decision support and experience management for ultrasonography”. In German Workshop on Experience Management 2003.

[139] Balaa ZE, Strauss A, Uziel P, Maximini K, Traphner R: “Fm-ultranet: a decision support system using case-based reasoning, applied to ultrasonography”. In Workshop on CBR in the Health Sciences, ICCBR 2003, 37–44.

[140] Perner P: “An architecture for a cbr image segmentation system”. Journal on Engineering Application in Artificial Intelligence 1999, 12(6):749–759.

[141] Bichindaritz I: “Solving safety implications in a case based decision-support system in medicine”. In Workshop on CBR in the Health Sciences, ICCBR 2003, 9-18.

[142] Schmidt R, Steffen D Gierl L: “Evaluation of a case-based antibiotics therapy adviser”. Artificial Intelligence in Medicine 2001, 462–466.

[143] Gierl L, Schmidt R: “Cbr in medicine. Case- Based Reasoning Technology, from Foundations to Applications” 1998, 273–297.

[144] Schmidt R, Pollwein B, Gierl L: “Experiences with case-based reasoning methods and prototypes for medical knowledge-based systems”. In Artificial Intelligence in Medicine 1999, 124–132.

[145] Costello E, Wilson DC: “A case-based approach to gene finding”. In Workshop on CBR in the Health Sciences, ICCBR 2003, 19–28.

[146] Montani S, Magni, P, Roudsari AV, Carson ER, Bellazzi R: “Integrating different methodologies for insulin therapy support in type 1 diabetic patients”. In Artificial Intelligence in Medicine 2001, 121–130.

[147] Perner P, Gunther T, Perner H: “Airborne fungi identification by case-based reasoning”. In Workshop on CBR in the Health Sciences, ICCBR 2003, 73–79.

[148] Evans-Romaine K, Marling C: "Prescribing exercise regimens for cardiac and pulmonary disease patients with cbr". In Workshop on CBR in the Health Sciences, ICCBR 2003, 45–52.

[149] Bichindaritz I, Sullivan K: "Generating practice cases for medical training from a knowledge-based decision-support system". In Workshop Proceedings ECCBR 2002, 3–14.

[150] Marling C, Whitehouse P: "Case-based reasoning in the care of alzheimer's disease patients". In Case-Based Research and Development, ICCBR 2001, 702–715.

[151] Davis G, Wiratunga N, Taylor B, Craw S: "Matching smarthouse technology to needs of the elderly and disabled". In Workshop on CBR in the Health Sciences, ICCBR 2003, 29–36.

[152] UMLS® Reference Manual, Introduction to the UMLS - http://www.ncbi.nlm.nih.gov/books/NBK9675 (accessed January 10, 2013).

[153] Cunningham H, Maynard D, Bontcheva K: "Text Processing with GATE (Version 6)". University of Sheffield Department of Computer Science 2011.

[154] Brill E: "A simple rule-based part of speech tagger". Proc 3rd Ann Conf on Applied Natural Language Processing 1992.

[155] Entrez Help - http://www.ncbi.nlm.nih.gov/books/NBK3837 (accessed January 10, 2013).

[156] Sayers E, Miller V: "Overview of the E-utility Web Service (SOAP)". In Entrez Programming Utilities Help - http://www.ncbi.nlm.nih.gov/books/NBK43082 (accessed January 10, 2013).

[157] Sayers E: "A General Introduction to the E-utilities". In Entrez Programming Utilities Help – http://www.ncbi.nlm.nih.gov/books/NBK25497 (accessed January 10, 2013).

[158] Hanson R, Tacy A: "GWT in Action, Easy Ajax with the Google Web Toolkit". Manning Publications 2007.

[159] Ajax: A New Approach to Web Applications - http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications (accessed January 10, 2013).

[160] The Apache Software Foundation, Licenses –

http://www.apache.org/licenses (accessed January 10, 2013).

[161] Cooper R, Collins C: “GWT in Practice”. Manning Publications 2008.

[162] Dewsbury R: "Google Web Toolkit Applications". Prentice Hall 2007.

[163] Making Remote Procedure Calls - Google Web Toolkit - Google Developers - https://developers.google.com/web-toolkit/doc/latest/tutorial/RPC?hl=de-DE (accessed January 10, 2013).

[164] Corlan AD: “Medline trend: automated yearly statistics of PubMed results for any query” - http://dan.corlan.net/medline-trend.html (accessed January 10, 2013).

[165] Maglott D, Ostell J, Pruitt KD, Tatusova T: “Entrez Gene: gene-centered information at NCBI”. Nucleic Acids Res 2011, 39:D52-D57.

[166] Wishart DS, Knox C, Guo AC, Eisner R, Young N, Gautam B, Hau DD, Psychogios N, Dong E, Bouatra S, Mandal R, Sinelnikov I, Xia J, Jia L, Cruz JA, Lim E, Sobsey CA, Shrivastava S, Huang P, Liu P, Fang L, Peng J, Fradette R, Cheng D, Tzur D, Clements M, Lewis A, De Souza A, Zuniga A, Dawe M, Xiong Y, Clive D, Greiner R, Nazyrova A, Shaykhutdinov R, Li L, Vogel HJ, Forsythe I: “HMDB: a knowledgebase for the human metabolome”. Nucleic Acids Res 2009, 37:D603-D610.

[167] Shah PK, Perez-Iratxeta C, Bork P, Andrade MA: “Information extraction from full text scientific articles: where are the keywords?”. BMC Bioinformatics 2003, 4:20.

[168] Entrez Gene Statistics – http://www.ncbi.nlm.nih.gov/projects/Gene/gentrez_stats.cgi (accessed January 10, 2013).

[169] HMDB - http://www.hmdb.ca/ (accessed January 10, 2013).

[170] UniProtKB/Swiss-Prot Release 2012_11 statistics – http://web.expasy.org/docs/relnotes/relstat.html (accessed January 10, 2013).

[171] UMLS - Metathesaurus Release Statistics – http://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus/release/statistics.html (accessed January 10, 2013).

[172] UMLS® Reference Manual, Metathesaurus - Rich Release Format (RRF) - http://www.ncbi.nlm.nih.gov/books/NBK9685 (accessed January 10, 2013).

[173] Jiang J, Conrath D: "Semantic similarity based on corpus statistics and lexical taxonomy". In Proceedings of the international conference on research in computational linguistics, 1997, 19–33.

[174] Markey MK, Lo JY, Tourassi GD, Floyd Jr CE: "Self-organizing map for cluster analysis of a breast cancer database". Artif Intell Med 2003, 27(2):113–27.

[175] Coulter DM, Bate A, Meyboom RH, Lindquist M, Edwards IR: "Antipsychotic drugs and heart muscle disorder in international pharmacovigilance: data mining study". BMJ 2001, 322(7296):1207–1209.

[176] Caviedes JE, Cimino JJ: "Towards the development of a conceptual distance metric for the UMLS". J Biomed Inform 2004, 37(2):77–85.

[177] Melton GB, Parsons S, Morrison FP, Rothschild AS, Markatou M, Hripcsak G: "Inter-patient distance metrics using SNOMED CT defining relationships". J Biomed Inform, 39(6):697–705.

[178] Apache Tomcat - Welcome! - http://tomcat.apache.org (accessed January 10, 2013).

[179] Elliott P, Andersson B, Arbustini E, Bilinska Z, Cecchi F, Charron P, Dubourg O, Kühl U, Maisch B, McKenna WJ, Monserrat L, Pankuweit S, Rapezzi C, Seferovic P, Tavazzi L, Keren A: "Classification of the cardiomyopathies: a position statement from the European Society Of Cardiology Working Group on Myocardial and Pericardial Diseases". Eur Heart J 2008, 29:270-6.

[180] Ahamad F, Seidman JG, Seidman CE: "The genetic basis for cardiac remodelling". Annu Rev Genomics Hum Genet 2005, 6:185-216.

[181] Meune C, Van Berlo JH, Anselme F, Bonne G, Pinto YM, Duboc D: "Primary prevention of sudden death in patients with lamin A/C gene mutations". N Engl J Med 2006, 354:209-210.

[182] Larizza C, Gabetta M, Milani G, Bucalo M, Mulas F, Nuzzo A, Favalli V, Arbustini E, Bellazzi R: "Supporting translational research on inherited cardiomyopathies through Information Technology". Methods of Information in Medicine 2013, *in press*.

[183] Roos M, Marshall MS, Gibson AP, Schuemie M, Meij E, Katrenko S, Van Hage WR, Krommydas K, Adriaans PW: "Structuring and extracting knowledge for the support of hypothesis generation in molecular biology". BMC Bioinformatics 2009, 10:S9.

[184] Spärck Jones K: "A statistical interpretation of term specificity and its application in retrieval". Journal of Documentation 1972, 28:11-21.

[185] Newman MEJ: "Mathematics of networks". The New Palgrave Encyclopedia of Economics 2008.

[186] Pasotti M, Klersy C, Pilotto A, Marziliano N, Rapezzi C, Serio A, Mannarino S, Gambarin F, Favalli V, Grasso M, Agozzino M, Campana C, Gavazzi A, Febo O, Marini M, Landolina M, Mortara A, Piccolo G, Viganò M, Tavazzi L, Arbustini E: "Long-term outcome and risk stratification in dilated cardiolaminopathies". J Am Coll Cardiol 2008, 52:1250–1260.

[187] Segagni D, Tibollo V, Dagliati A, Zambelli A, Priori SG, Bellazzi R: "An ICT infrastructure to integrate clinical and molecular data in oncology research". BMC Bioinformatics 2012, 13(4):S5.

[188] Segagni D, Gabetta M, Tibollo V, Zambelli A, Priori SG, Bellazzi R: "ONCO-i2b2: improve patients selection through case-based information retrieval techniques". Lecture Notes in Computer Science 2012, 7348:93-99.

[189] Mate S, Bürkle T, Köpcke F, Breil B, Wullich B, Dugas M, Prokosch HU, Ganslandt T: "Populating the i2b2 database with heterogeneous EMR data: a semantic network approach". Stud Health Technol Inform 2011, 169:502-506.

[190] InfoVis JavaScipt Toolkit - http://www.thejit.org (accessed January 10, 2013).