



UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Informatica

Corso di Laurea Triennale in Informatica

TESI DI LAUREA

**SPERIMENTAZIONE DI UN SISTEMA PER
IL SUPPORTO ALLA DIAGNOSI DI
MALATTIE CRITICHE IN AMBITO
OSPEDALIERO**

RELATORI

Prof. Vincenzo Deufemia

Dott.ssa Loredana Caruccio

CANDIDATO

Dario Di Dario

Matricola: 0512104758

Anno Accademico 2018-2019

*A nonno Manfredi e nonna Teresa: le stelle che illuminano il mio
cammino...*

Abstract

Sin dalla nascita dei primi algoritmi e quindi, dei primi prototipi software atti a risolvere una qualsivoglia classe di problemi, c'era il bisogno di testare l'effettiva funzionalità del software stesso, dimostrando (non solo attraverso calcoli scientifici) le varie funzionalità di quest'ultimo.

Questa fase, in Ingegneria del Software è detta: *collaudo* o *testing* si pone l'obiettivo di verificare e validare di quanto il prodotto software sviluppato soddisfi i requisiti individuati dall'analisi.

L' obbiettivo di questa tesi è quindi quello di *collaudare* in maniera piu accurata possibile, il tool che viene classificato come un *CDS* (*Clinical Decision Support System*) che mira a supportare i medici nella diagnosi di malattie critiche e a ridurre il più possibile la latenza della loro corretta identificazione.

Per riuscire a sviluppare questo progetto di tesi è stato necessario comprendere a fondo il funzionamento del software, come i parametri in input devono essere strutturati per ottenere una corretta elaborazione degli artefatti e come i risultati potessero essere visibili per essere studiati e per trarne delle conclusioni.

Sono state così prese in considerazione un certo numero di cartelle cliniche di pazienti reali (che per ovvi motivi di privacy non saranno mai citati in questo elaborato) che in passato hanno presentato casi di *SIRS* e di *SEPSI*. Questo tipo di *CDS* analizza i dati clinici in formato digitale tramite tecniche di *Natural Language Processing (NLP)* al fine di permettere l'estrazione delle informazioni contenute nelle cartelle cliniche. Le informazioni estratte, successivamente, vengono modellate e validate attraverso linguaggi di rappresentazione semi-strutturati, quali l'*XML* (*eXtensible Markup Language*) e *XSD* (*XML Schema Definition*) per poi essere trattati e rappresentati in termini di concetti di un'ontologia. Ovviamente si precisa che il caso clinico della *SEPSI* è soltanto un punto di partenza, ricordando che il tool può essere esteso a qualsiasi altra malattia per la quale vengono definiti dei parametri da analizzare per la diagnosi della stessa.

Contents

Indice	ii
Elenco delle figure	iv
Elenco delle tabelle	vi
1 Introduzione	1
1.1 Motivazioni e Obiettivi	1
1.2 Struttura della tesi	2
2 Stato dell'arte	3
2.1 NLP come strumento di supporto in ambito sanitario	3
2.2 Sistemi NLP per il riconoscimento di malattie	5
2.3 Analisi della letteratura	8
2.4 Confronto	10
3 Concetti preliminari	12
3.1 Natural Language Processing	12
3.2 Ontologie	13
4 L'approccio metodologico	15
4.1 Architettura del sistema	15
4.1.1 Fase di pre-processing	16
4.1.2 Fase di estrazione e analisi delle informazioni	19

4.1.3	Creazione e validazione del file XML	22
4.1.4	Popolamento ontologia	23
4.1.5	Fase di inferenza	23
4.1.6	Presentazione del tool	24
4.1.7	L'istallazione del tool e le sue problematiche	25
5	Valutazione Sperimentale	29
5.1	La Sepsi	29
5.2	La sperimentazione del tool	31
5.2.1	Breve introduzione all'applicativo Protégé	37
5.3	Risultati ottenuti dall'elaborazione del software	37
5.4	La controprova	43
6	Conclusioni e sviluppi futuri	48
6.1	Conclusioni	48
Bibliography		50
Ringraziamenti		53

List of Figures

2.2	<i>Secondo modello NLP-CDS</i>	5
2.3	<i>Funzionamento DNorm</i>	7
2.4	<i>Sistema NLP per l'estrazione dei problemi riportati in una cartella clinica</i>	8
4.1	<i>Architettura del sistema</i>	16
4.2	<i>Informazioni non strutturata</i>	19
4.3	<i>Informazioni strutturali</i>	19
4.4	<i>Esempio di analisi del diario clinico</i>	20
4.5	<i>Modellazione di un parametro in XML</i>	22
4.6	<i>Estratto visuale del file XSD utilizzato per la validazione</i>	23
4.7	<i>Processo grafico per l'analisi della cartella clinica in formato digitale</i>	24
4.8	<i>Processo grafico per eseguire il processo inferenziale</i>	25
4.9	<i>Una parte delle librerie esterne utilizzate</i>	26
4.10	<i>Path statico</i>	26
4.11	<i>Parametri per la VM</i>	28
5.1	<i>Relazione tra Sirs, Sepsis ed Infezione</i>	30
5.2	<i>Diario clinico illegibile</i>	32
5.3	<i>Gestione dati illegibili</i>	33
5.4	<i>Scelta formato della cartella clinica</i>	34
5.5	<i>Il processo è stato completato con successo</i>	34
5.6	<i>Struttura del file XML con i dati estrapolati</i>	35
5.7	<i>Esecuzione del processo inferenziale</i>	35

5.8	<i>Messaggio di completamento</i>	36
5.9	<i>Interfaccia grafica di Protégé</i>	37
5.10	<i>Differenze tempi esecuzione</i>	40
5.11	<i>Grafico delle caratteristiche</i>	40
5.12	<i>Relazione tra tempi OWL e numero giorni Sirs</i>	41
5.13	<i>Tempi esecuzione VS conteggio caratteri</i>	42
5.14	<i>Resoconto pazienti</i>	43
5.15	<i>Parametri con esito Sirs</i>	45
5.16	<i>Modifica parametri</i>	46
5.17	<i>Nessun esito positivo per Sirs.</i>	47

List of Tables

2.1	Framework e tool di IE	11
5.1	Metriche relative ai pazienti	38
5.2	Metriche aggiuntive	39
5.3	Medie dei tempi di esecuzione	39

CHAPTER 1

Introduzione

1.1 Motivazioni e Obiettivi

Fin dalla sua nascita l'*Intelligenza artificiale (AI)* è una tecnologia informatica che ha rivoluzionato il modo con cui l'uomo interagisce con la macchina.

Grazie all'intelligenza artificiale è possibile rendere le macchine in grado di compiere azioni e "ragionamenti" complessi, imparare dagli errori, e svolgere funzioni fino ad oggi esclusive dell'intelligenza umana. Oggi in Italia e nel mondo, l'intelligenza artificiale viene utilizzata in azienda e non solo, per svolgere compiti che all'uomo richiederebbero molto tempo, ma è applicato anche ad ulteriori ambiti, quali i videogiochi. Un algoritmo di intelligenza artificiale permette ai personaggi, gli ambienti, alle storie di evolversi secondo il comportamento del giocatore, creando situazioni sempre nuove ed imprevedibili.

Vista la potenza, le capacità e le possibilità offerte dall'AI insieme alla crescente quantità di dati digitali a disposizione, recentemente sono nati nuovi ambiti di ricerca, riguardanti diverse discipline, che spaziano dalla domotica fino ad arrivare ad applicare l'AI al mondo della sanità e dell' Health-Care [1]. Ad esempio prendiamo in considerazione uno degli ultimi dispositivi di casa *Apple*: *iWatch serie 5* il quale integra un sensore di altissima precisione in grado di monitorare costantemente i battiti cardiaci, rilevando e notificando casi anomali in cui il battito diventa irregolare; così facendo si cerca di supportare l'attività dei medici nel processo di diagnosi e cura di malattie critiche. Infatti il processo diagnostico è un processo assai complicato, che al contempo richiede una buona preparazione e prontezza nel diagnosticare

malattie, in modo tale da riuscire a guadagnare quanto più tempo possibile avendo maggiori probabilità di salvare il paziente. Ovviamente, i computer possono far molto in questo senso, vista la capacità di analizzare dati ad una elevatissima velocità dei tempi di calcolo. Di fatto, questo non vuole sostituire le importanti attività di diagnosi dei medici, piuttosto vuole fornire un supporto ad essi, sfruttando le alte capacità dei calcoli delle macchine.

A tale scopo si propone il presente lavoro, che ha come obiettivo la sperimentazione del tool basato sulla semantica di dati presenti in cartelle cliniche, che mira a fornire uno strumento di supporto utile alla diagnosi tempestiva di malattie critiche in ambito ospedaliero. Tale approccio, che fonda le sue basi partendo dalle cartelle cliniche reali che vengono studiate per poi essere trascritte in un formato digitale, si focalizza sull'identificazione di indicatori parametrici al fine di allertare i medici sull'ipotesi che possa essere diagnosticata una particolare malattia e/o patologia.

Questo tipo di sperimentazione mira a valutare l'accuratezza del tool, al fine di comprendere se potrebbe essere introdotto in ambiti come il pronto soccorso, dove la numerosità dei pazienti porta all'impossibilità per i medici e/o infermieri di effettuare un monitoraggio costante sui pazienti che posso mostrare patologie con elevate velocità di manifestazioni e un tasso di mortalità molto alto. Uno scenario di questo tipo è rappresentato dalle infezioni come la *Sepsi*.

1.2 Struttura della tesi

Il resto di questo lavoro di tesi è strutturato come segue: nel **Capitolo 2**, si presenta una panoramica dello stato dell'arte relativo all'analisi di dati clinici attraverso tecniche di *NLP* [7]. Il **Capitolo 3** presenta i dettagli sul processo di analisi di questa proposta e l'architettura del tool utilizzato. Il **Capitolo 4** spiega in modo dettagliato i passi e le operazioni necessarie per la sperimentazione delle cartelle cliniche, commentando successivamente i risultati ottenuti. Infine il **Capitolo 5** vengono fornite considerazioni conclusive e possibili sviluppi futuri.

CHAPTER 2

Stato dell'arte

In questo capitolo si propone una panoramica dello stato dell'arte, in merito all'analisi di dati clinici appartenenti a pazienti reali, tramite tecniche NLP. Saranno presentati esclusivamente gli approcci simili a quelli della metodologia proposta. Infine, si procederà ad effettuare un'analisi tra gli approcci e la proposta descritta.

2.1 NLP come strumento di supporto in ambito sanitario

I dati clinici, sono definiti sottoforma di testo non strutturato, ciò significa che non seguono una struttura precisa e che quindi l'analisi stessa può diventare ostica, motivo per cui, ci si pone l'obiettivo di effettuare un'estrazione delle informazioni utili. Ciò che si ottiene dai processi NLP è l'estrazione delle informazioni, abilitando il loro utilizzo nei sistemi di supporto alle decisioni cliniche [3].

Un CDS (in inglese *Clinical Decision Support System*) è considerato come un software di supporto alle decisioni, che permette di aumentare l'efficacia delle analisi in quanto fornisce supporto a tutti coloro che devono prendere decisioni "strategiche". La funzione principale di un CDS è quella di estrarre in poco tempo e in modo versatile le informazioni utili ai processi decisionali. Con l'avvento dei CDS si riscontrano miglioramenti delle decisioni cliniche di circa il 60%. I CDS possono essere caratterizzati in vari modi, sulla base di differenti aspetti:

- **access:** integrato in un altro sistema o stand-alone;

- **setting:** usato in ambulatorio o in regime di ricovero;
- **task:** mirato ad uno specifico compito clinico o amministrativo, come diagnosi o controllo di qualità;
- **scope:** generale o destinato ad un compito specifico;
- **timing:** utilizzato prima, durante o dopo la decisione clinica;
- **output:** attivo (invio di promemoria ad esempio) o passivo;
- **implementazione:** basato sulla conoscenza o statistico.

I moduli NLP e CDS hanno relazioni ben definite che risultano essere descritte tramite la combinazione di tre assi:

- **integration:** se il sistema NLP è integrato col CDS o accoppiato ad esso;
- **government:** se il flusso d'elaborazione dei dati del sistema NLP è gestito e controllato da un CDS o meno;
- **specialization:** se il sistema NLP è stato sviluppati per un compito specifico o come un tool generale che può essere personalizzato per risolvere diversi compiti.

Si pensi che, a seconda delle particolari combinazioni NLP-CDS lungo i vari assi sopra descritti possono portare a diversi modelli, due dei quali sono rappresentati in *Figura 2.1* e *Figura 2.2*.

I modelli sopra rappresentati vengono descritti nel primo caso, come la differenza strutturale nel caso in cui un *NLP system* sia sviluppato per concepire un solo compito specifico, che risulta essere accoppiato un modulo CDS. (*Figura 2.1*) e un sistema NLP-CDS che risulta essere integrato, self-governed e multi tasking (*Figura 2.2*).

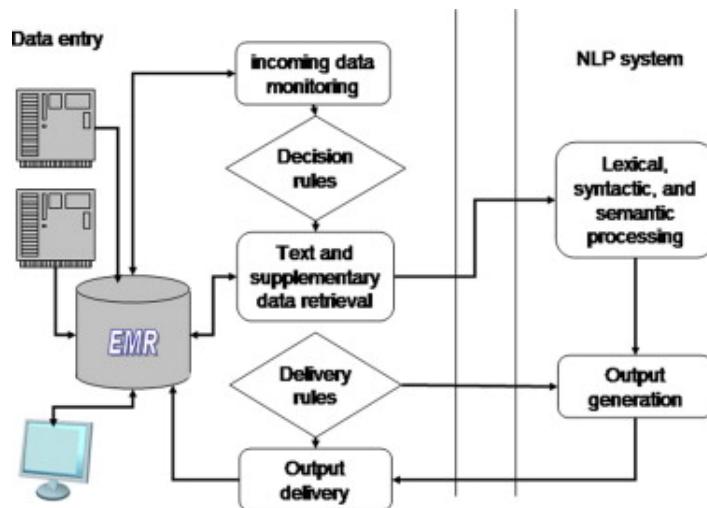


Figure 2.1: Primo modello NLP-CDS

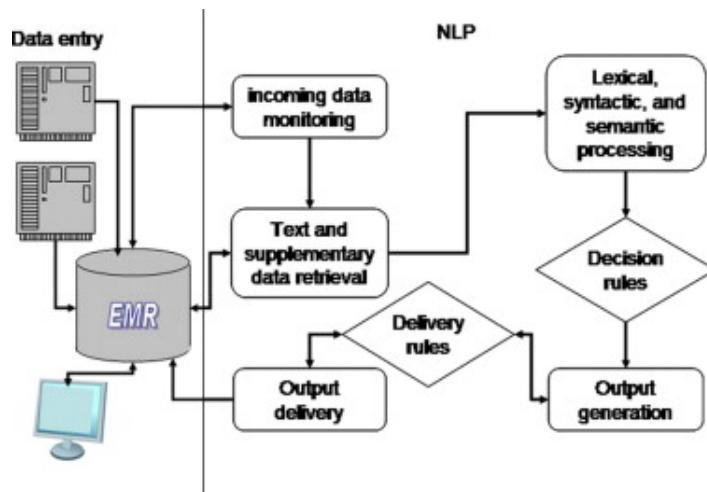


Figure 2.2: Secondo modello NLP-CDS

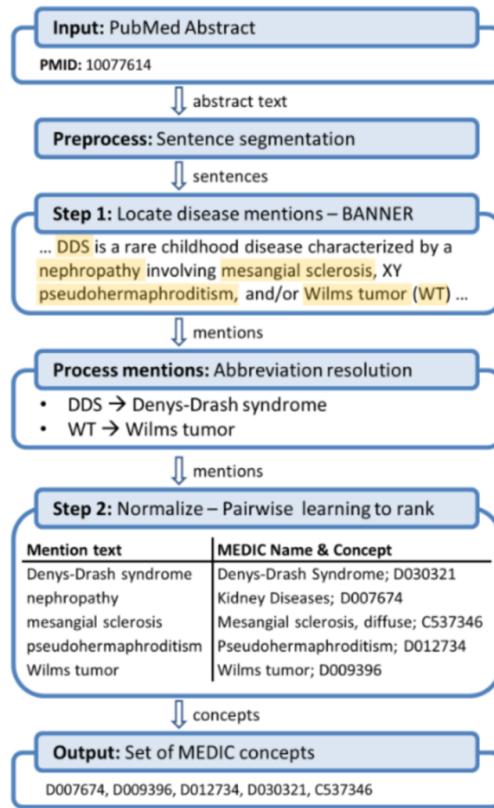
2.2 Sistemi NLP per il riconoscimento di malattie

In letteratura, sono stati definiti diversi sistemi *NLP* per il riconoscimento di malattie. Ad esempio, in [10] si presenta un sistema di *NLP* per il riconoscimento automatico di casi di *malattia periferica arteriosa degli arti inferiori (PAD)* a partire da note cliniche. L'algoritmo *NLP* è stato applicato ad un dataset proveniente dalla clinica Mayo di Rochester, nel Minnesota. Il dataset, creato a giugno 2015 contiene 1569 pazienti (di cui 806 avevano la *PAD* e 763 no), è stato casualmente suddiviso in training set (935 pazienti, 300.364 note cliniche, 479 casi di *PAD*, 456 casi senza *PAD*) e test set (634 pazienti, 212.047 note cliniche, 327 casi di *PAD*, 307 casi senza *PAD*). L'algoritmo è stato addestrato sul training set e validato sul test set; le prestazioni dell'algoritmo sono poi state confrontate con gli algoritmi che effettuano riconoscimento di codici delle malattie con i metodi standard di riferimento per il riconosci-

mento di *PAD*, ottenendo migliori risultati. L'algoritmo ha due componenti principali: *text processing* e *patient classification*. La componente di *text processing* trova i concetti relativi alla *PAD* attraverso MedTagger e li mappa a delle categorie specifiche. Poi, la componente di *patient classification* usa un insieme di regole predefinito per classificare lo stato di ogni individuo. Una prima versione dell'algoritmo utilizzava tutte le sezioni contenute nelle note cliniche; poi, al fine di migliorare le prestazioni, alcune sezioni sono state escluse dall'analisi.

Alla base dell'algoritmo per la *PAD*, lo si ha esteso ulteriormente anche per il riconoscimento dell'ischemia degli arti (*CLI*) che, risulta essere una complicazione della *PAD*, come presentato in [11].

In [15] è stato presentato un sistema, atto all'identificazione automatica dei disturbi presenti in una narrativa clinica utilizzando una versione di *Banner* [4], un metodo statistico per il riconoscimento di entità e successivamente migliorata attraverso l'aggiunta di concetti *UMLS* (Unified Medical Language System) individuati grazie a *MetaMap*, e di *DNorm* [14] (Figura 2.3). In particolare *DNorm* utilizza il *machine learning* per normalizzare i nomi dei disturbi e il *pairwise learning* per addestrare una funzione di similarità tra i concetti trovati nel testo clinico ed un vocabolario predefinito.

**Figure 2.3: Funzionamento DNorm**

Successivamente, in [5] è stata presentata un'applicazione di NLP per ottenere le estrazioni dei problemi clinici a partire da note cliniche. Una nota clinica è definita come un riferimento ad un cartella clinica in formato elettronico, analizzandone il testo sperando di ottenere informazioni rilevanti. La parte NLP del sistema utilizza *MetaMap* [12] nella sua vecchia versione Java (*MMTx*), per mappare il testo analizzato ad un sottoinsieme di concetti *UMLS* (*Unified Medical Language System*) [13], e l'algoritmo *NegEx* [22], per l'estrazione di 80 problemi medici riguardanti la medicina e la chirurgia cardiovascolare. Lo scopo del progetto è stato quello di automatizzare il processo di creazione e manutenzione della lista dei problemi in modo chiaro e conciso per supportare e facilitare il lavoro del medico. Il processo è schematizzato in *Figura 2.4*.

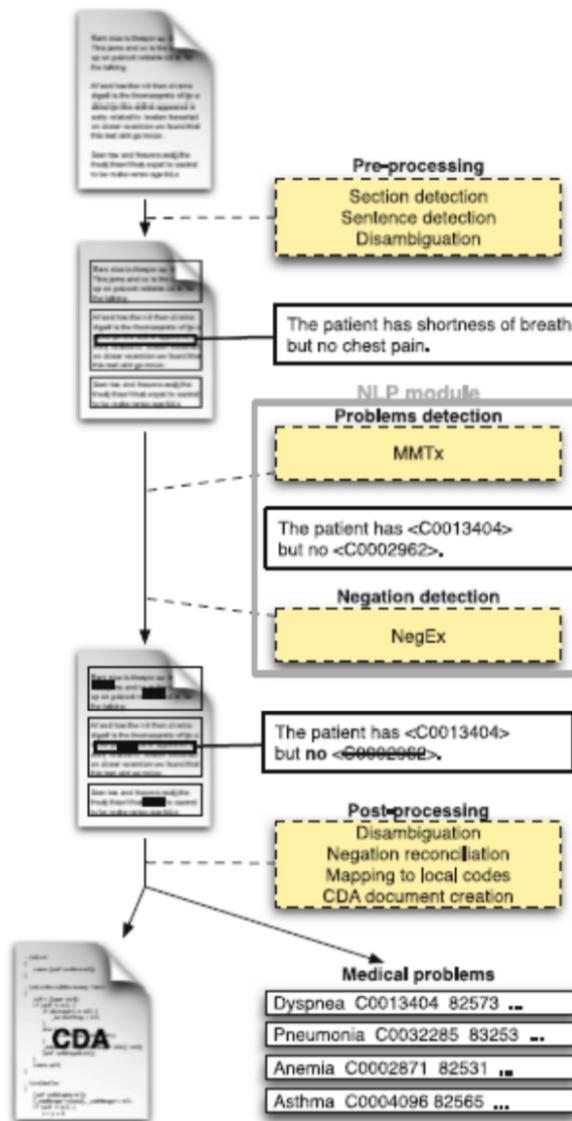


Figure 2.4: Sistema NLP per l'estrazione dei problemi riportati in una cartella clinica

2.3 Analisi della letteratura

La molteplicità delle applicazioni di *NLP* e *CDS* descritte in letteratura, ha portato alla loro analisi e comparazione in diverse literature review che mirano alla caratterizzazione e alla messa in evidenza dei punti di forza e di debolezza dei singoli approcci, quando vengono messi a confronto. Nei seguenti articoli [7], [28], [8], [27] si nota una review riguardo alla letteratura in merito alle metodologie per l'analisi e l'estrazione dei dati.

In [7] è stata condotta un'analisi della letteratura per identificare i sistemi di *NLP* in ambito clinico. Dopo una fase di selezione, 86 articoli sono stati considerati, ottenendo 71 sistemi. Tra

i sistemi trovati la tecnica più comune utilizzava un approccio *NLP ruled-based* (33 sistemi), mentre alcuni (solo 4) utilizzavano un approccio di machine learning "puro"; in generale, i sistemi progettati e studiati, si basavano su metologie che risolvessero uno specifico obiettivo.

In [28] viene effettuato un resoconto sugli ambiti di ricerca interessati dall'analisi dei dati clinici (*diagnosis categorization, novel phenotype discovery, clinical trial screening, pharmacogenomics, drug-drug interaction, adverse drug events, genome-wide association studies and phenome-wide association studies*) e sulle tecniche utilizzate per effettuare tali analisi (*keyword search and rule-based system, supervised learning system, unsupervised system, deep learning*). Si nota che i sistemi progettati con un approccio *keyword search* e *regole* offrono un'indice di accuratezza maggiore; tuttavia, la costruzione manuale delle regole e delle keyword genera problemi di generalizzabilità e scalabilità per questi metodi. Sistemi basati sul *supervised learning* sono accurati e facili da addestrare e testare; Essi, però, richiedono un'abbondante mole di lavoro per etichettare i dati di training. I metodi basati sul *deep learning* però, sono sempre più utilizzati insieme all'*unsupervised learning*, grazie al meno sforzo umano che richiedono. Da quanto appena descritto, si cocerne che non esiste un metodo che sia predominante, né che uno sia "migliore" di un altro, poichè bisogna considerare la forte dipendenza tra le prestazioni e i dati della sorgente in input.

In [8] si prende visione delle principali tecniche di *data mining* e delle misure statistiche da prendere in considerazione per valutare le prestazioni di una tecnica (*precision, recall, f-score, accuracy, sensitivity e specificity*). Successivamente, ci si concentra sul *text mining*, a cui è affidato il processo di estrazione delle informazioni da testo strutturato o non, e sulle principali tecniche utilizzate. Viene reso noto un resoconto riguardante i dati clinici, la loro classificazione a livello internazionale (attraverso l'uso degli ICD *codes*) e le tecniche di *text mining* applicate a tali dati. Inoltre, in tale review sono indicati i molti problemi da tenere in considerazione nell'elaborare i dati clinici: testo strutturato e non, termini molto tecnici, abbreviazioni, acronimi, errori di scrittura, mancanza di punteggiatura, differenti tipi di dati, problemi di privacy dei pazienti, ecc. Infine, vengono presentati alcuni dei principali tool utilizzati per classificare i dati clinici e per supportare le decisioni dei medici.

In [27] si presenta un'introduzione all'utilizzo nell'ambito della ricerca di *EHR* (electronic health records) e nel contempo vengono paragonati gli articoli relativi all'*NLP*. Quindi, viene effettuata un'analisi sui principali tool di *IE* (*Information Extraction*), sui metodi prevalente-

mente utilizzati (*rule-based* e *machine learning*), e sui principali casi di utilizzo in base a parte della letteratura (scritta in inglese) pubblicata tra il 2009 e settembre 2016 in ambito di *clinical medicine, informatics* e *computer science*. La *Figura 2.9* seguente mostra i tool di IE (*Figura 2.9*). Come è possibile notare, la maggior parte dei sistemi utilizza un approccio *rule-based*, ma non è da sottovalutare il relativo spostamento verso approcci che siano basati su *machine-learning*.

2.4 Confronto

Come da titolo, questa sezione svilupperà un confronto mostrandone differenze e similitudine tra l'approccio utilizzato e gli algoritmi sopra presentati. Il tool utilizza i dati che provengono da un'unica cartella clinica per cercare un ipotetico riscontro di sepsi, la fonte dunque, non proviene da un dataset. I dati quindi, sono estratti una un formato *.docx* o *.pdf*. Oltre tutto, vengono estratte tutte le informazioni che risultano essere significative (informazioni sul paziente, sul ricovero, anamnesi, sintomi, terapie, esami del sangue, ecc.), mentre negli approcci sopra descritti, si prendono le informazioni relative alla PAD. Il fine è quello di ottenere un'estrazione dei dati quanto più precisa possibile, senza tralasciare alcuna informazione. L'estrazione delle informazioni segue un approccio *rule-based* attraverso l'utilizzo di *MetaMap Lite* [21], una versione più leggera e veloce di *MetaMap* che consente di mappare il testo a concetti *UMLS*, e di *regular-expression*. Le informazioni estratte vanno poi a formare un file *XML* che, validato secondo uno specifico schema definito in *XSD*, costituisce la sorgente informativa per un'ontologia su cui vengono effettuate le analisi. Il tutto è stato fatto per rendere l'approccio quanto più estensibile possibile, permettendo ai futuri sviluppatori di occuparsi solo dei parser verso *XML* e della definizione di nuove regole di diagnosi per il riconoscimento di altre malattie. Infine, bisogna sottolineare come la proposta non si fermi alla ricerca delle malattie riportate nella cartella clinica, ma effettui, in tempo reale, un'analisi dei dati estratti per diagnosticare una malattia che potrebbe anche non essere per nulla menzionata.

Table 2.1: Framework e tool di IE

Name	Description
<i>Framework UIMA</i>	Software framework for the analysis of unstructured contents like: text, video and audio data.
GATE	Java-based open-source software for various NLP task such as information extraction and semantic annotation
Protégé	Open-source ontology editor and framework for building intelligent system
Tools cTAKES	Open-source NLP system based on UIMA framework for extraction of information from electronic health records unstructured clinical text
MetaMap	Nation institute of health (NIH)-developed NLP tool that maps biomedical text to UMLS concepts
MedLee	NLP system that extracts, structures, and encodes clinical information from narrative clinical notes
KnowledgeMap Concept Indexer	NLP system that identifies biomedical concepts and maps them to UMLS concepts
HTTEEx	Open-source NLP tool built on top of the GATE framework for various tasks such as principal diagnoses extraction and smoking status extraction
MedEx	NLP tool used to recognize drug names, dose, route, and frequency from free-text clinical records
MedTagger	Open-source NLP pipeline based on UIMA framework for indexing based on dictionaries, information extraction, machine learning-based named entity recognition from clinical text

CHAPTER 3

Concetti preliminari

In questo capitolo saranno presentate le tecnologie che consentono il processo di automazione delle diagnosi e verrà costruita una panoramica sul Natural Language Processing (NLP) e le ontologie.

3.1 Natural Language Processing

L' **NLP** (*Natural Language Processing*) è un campo di ricerca interdisciplinare che abbraccia **informatica, intelligenza artificiale e linguistica** il cui scopo è quello di sviluppare algoritmi che siano in grado di analizzare, rappresentare e quindi "comprendere" il linguaggio naturale, scritto o parlato, in maniera similare o addirittura migliore rispetto agli esseri umani. Tale "comprensione" è determinata dal capire, e quindi essere poi in grado di usare, il linguaggio a varie granularità, dalle parole, in relazione al loro significato ed alla appropriatezza d'uso rispetto ad un contesto, fino alla grammatica ed alle regole di strutturazione delle frasi. Nonostante i notevoli passi avanti in questa area di ricerca, bisogna riconoscere come i problemi affrontati in questi processi siano abbastanza complessi.

I moderni sistemi di *NLP* fondano le loro basi sul *machine learning*, in cui si utilizza l'inferenza statistica per apprendere automaticamente le regole del linguaggio, analizzando numerose quantità di frasi. Gli ambiti in cui l'*NLP* è applicato, non sono pochi, si pensi alle numerose piattaforme presenti sul web, un esempio lampante è messo a disposizione dall'*e-commerce*: **Wish**. La piattaforma, consente di iniziare una vera e proria *chat* gestita in maniera automatica,

senza che le risposte siano date da un operatore fiscamente presente. Oppure, basti pensare al più famoso motore di ricerca *Google* che fornisce un servizio di traduzione automatica (*Google translate*), insomma, gli ambiti in cui l'intelligenza artificiale e del NLP sono svariati e più vicini di quanto sembri.

Il processo di elaborazione del linguaggio viene definito in più fasi, che sono:

- **analisi lessicale:** scomposizione di un'espressione linguistica in *token* (parole);
- **analisi grammaticale:** definito come il procedimento per identificare la categoria lessicale di ogni parola nel contesto nel quale è usata.
- **analisi sintattica:** arrangiamento dei token in una struttura sintattica (albero delle dipendenze).
- **analisi semantica:** assegnazione di un significato alla struttura sintattica e, dunque all'espressione linguistica.

Ci si accorge, dunque che effettuare un determinato tipo di elaborazione diventa ostico, motivo per cui esiste un insieme integrato di tool di analisi del linguaggio, prende il nome di **Stanford CoreNLP**. Esso è stato progettato per essere altamente flessibile ed estensibile, permettendo di scegliere quali funzionalità utilizzare a secondo delle necessità. Un'altra caratteristica da non sottovalutare è quella che consente di aggiungere tool di terze parti o personalizzati.

Alcune funzionalità messe a disposizione sono:

- **tokenize:** permette la suddivisione del testo in "token" (parole)
- **ssplit:** permette la divisione di un documento in frasi
- **lemma** che permette di riportare ogni parola alla propria forma base

e molte altre funzionalità che possono essere collegate tra loro. Si pensi ad esempio che per attuare la divisione di un documento in frasi, bisogna prima ottenere la suddivisione del testo in *token*, rispettivamente **ssplit** non può essere utilizzato senza prima applicare **tokenize**.

3.2 Ontologie

Un'ontologia [24] è una rappresentazione formale, condivisa ed esplicita di una concettualizzazione di un dominio di interesse. Più in dettaglio, si tatta di una teoria assiomatica

del primo ordine esprimibile in una logica descrittiva. Il termine *ontologia formale* è entrato in uso nel campo dell'*intelligenza artificiale* e della rappresentazione della conoscenza, per descrivere il modo in cui diversi schemi vengono combinati in una struttura dati contenente tutte le entità rilevanti e le loro relazioni in un dominio.

Per essere utili, le ontologie devono essere espresse in una notazione concreta. Un "linguaggio per le ontologie" è un linguaggio formale con cui viene costruita un'ontologia. Esistono diversi linguaggi, proprietari o basati su *standard* per la definizione delle ontologie, come:

- MOF e UML
- OWL
- OBO
- OntoUML

Quello che prediamo in considerazione è l'*OWL* (Web Ontology Language) è un linguaggio di markup per rappresentare esplicitamente significato e semantica di termini con vocabolari e relazioni tra gli stessi. Come vedremo nei capitoli successivi, un editor molto utilizzato per la gestione delle ontologie è *Protégé*, mentre per quanto riguarda la gestione di ontologie attraverso Java, si utilizza *Apache Jena*. Un vantaggio importante che si ottiene nella costruzione di applicazioni basate su ontologie è la possibilità di utilizzare i cosiddetti *reasoner* per derivare nuove informazioni vere riguardo i concetti che si stanno modellando. *Apache Jena* oltre a mettere a disposizione un numero predefinito di reasoner, consente anche di crearne dei propri in maniera semplice.

CHAPTER 4

L'approccio metodologico

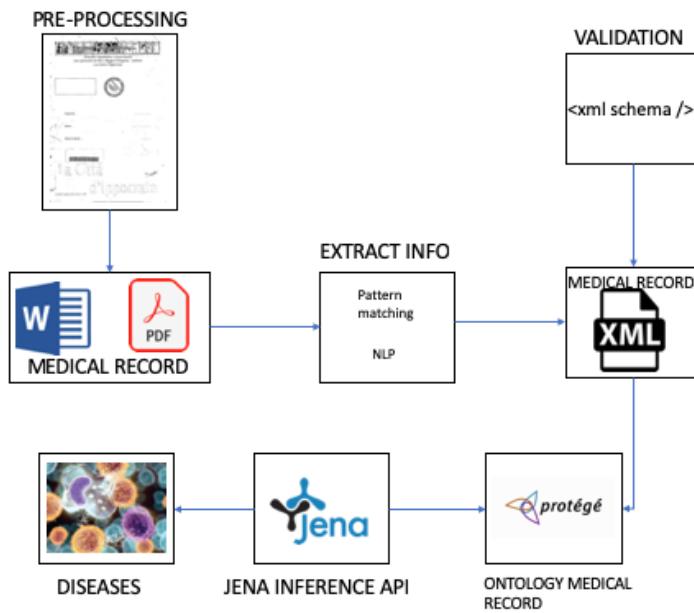
Il presente capitolo tratterà l'approccio metodologico adottato per permettere l'automatizzazione del processo di diagnosi. Tale approccio, come già introdotto, ha portato allo sviluppo di un tool che, partendo dai dati di una cartella clinica reale trascritti in un formato digitale e facendo uso delle tecnologie presentate nel precedente capitolo, cerca di identificare alcuni indicatori parametrici utili alla diagnosi tempestiva di una particolare malattia e/o patologia.

Per prima cosa, però, sarà presentata l'architettura del sistema sviluppato e, in seguito, seguendo il flusso d'elaborazione del sistema, si presenteranno le varie fasi e i rispettivi componenti in maniera dettagliata, instradando anche le modalità di installazione e i vari problemi che possono verificarsi durante tale fase.

4.1 Architettura del sistema

L'architettura del sistema sviluppato è presentata in *Figura 4.1*. Analizzandola visivamente è possibile scaturire le varie fasi del processo, i diversi componenti che la compongono, i dati che si ricevono in input e cosa si cerca di ottenere in output.

Successivamente si descriveranno passo dopo passo le funzionalità di ogni modulo .



La cartella clinica

Secondo le linee guida del Ministero della Salute 17 giugno 1992 inerenti alla compilazione, alla codifica e alla gestione della scheda di dimissione ospedaliera istituita ex DM 28.12.1991, la cartella clinica viene definita come: "lo strumento informativo individuale finalizzato a rilevare tutte le informazioni anagrafiche e cliniche significative relative ad un paziente e ad un singolo episodio di ricovero. Ciascuna cartella clinica ospedaliera deve rappresentare l'intero episodio di ricovero del paziente nell'istituto di cura: essa, conseguentemente, coincide con

la storia della degenza del paziente all'interno dell'ospedale. La cartella clinica ospedaliera ha così inizio al momento dell'accettazione del paziente in ospedale, ha termine al momento della dimissione del paziente dall'ospedale e segue il paziente nel suo percorso all'interno della struttura ospedaliera." [23].

La cartella clinica è un documento atto alla trascrizione e alla raccolta delle informazioni attinenti alle singole persone ricoverate, attraverso il quale vengono prese decisioni in merito allo stato di salute del paziente. I punti salienti dei contenuti che ne descrivono l'importanza sono riportati successivamente:

- **Generalità del paziente;**
- **motivo del ricovero;**
- **regime di ricovero;**
- **data e struttura di ammissione;**
- **provenienza del paziente;**
- **anamnesi;**
- **esame obiettivo;**
- **referti di esami diagnostici e specialistici;**
- **terapia;**
- **consensi e dichiarazioni di volontà;**
- **decorso della malattia;**
- **data e struttura di dimissione.**

Inoltre si menzionano i requisiti che bisogna rispettare per la compilazione di qualsiasi certificazione:

- **La chiarezza**, secondo cui il contenuto deve poter essere comprensibile anche per persone non esperte;
- **la veridicità**, secondo cui le segnalazioni effettuate e menzionate sulla cartella, devono essere conformi con quanto obiettivamente constatato;

- **la rintracciabilità**, ovvero la possibilità di risalire a tutte le attività agli esecutori, ai materiali e ai documenti dell'episodio di ricovero;
- **l'accuratezza** relative ai dati ed alle informazioni prodotte;
- **la pertinenza** ovvero la correlazione delle informazioni riportate in tabella rispetto alle esigenze informative definite;
- **la completezza** ovvero l'inserimento in cartella di tutti gli elementi che la compongono.

La cartella clinica è ricca di documenti, ognuno dei quali è corredata da informazioni specifiche e molto tecniche. Focalizziamoci sul tipo d'informazione contenuta all'interno di tali documenti. I vari documenti possono contenere due tipi di informazioni:

- **Informazioni strutturate**: dalla parola stessa, si comprende rapidamente che si sta parlando di un determinato tipo di informazioni avente una struttura prefissata di tipo *chiave-valore* (utilizzate per descrivere i dati del paziente in fase di ricovero) e dalle informazioni tabellari (utilizzate per le analisi di laboratorio);
- **Informazioni non strutturate**: sono rappresentate principalmente dal *diario clinico*, uno dei documenti più rilevanti della cartella clinica che mira a riportare, in maniera testuale, tutte le variazioni delle condizioni cliniche del paziente rispetto all'inquadramento clinico iniziale, le eventuali modifiche apportate al piano di cura, i risultati delle terapie praticate e così via. Tutte le informazioni nel diario clinico sono corredate da un'annotazione temporale che ne riporta la data e l'ora.

Nonostante il primo requisito descriva in modo chiaro e conciso che le cartelle cliniche devono poter essere comprese anche da persone non esperte, ma, essendo rappresentate principalmente da testo, sono sottoposte a problemi di ambiguità del linguaggio, inoltre, generalmente, sono scritte in maniera abbastanza veloce, lasciando dunque, ampio spazio agli errori grammaticali e lessicali, causando così ulteriori difficoltà durante la fase di analisi. Una volta che le informazioni hanno superato la fase di ambiguità, si passa alla fase di trascrizione delle stesse informazioni (in inglese) in formato *.docx* e da quest'ultimo, viene generato infine un *.pdf* del file trascritto. Nelle successive due figure, sono rappresentate le informazioni strutturali e non, dei file creati.

11/03/16	7.10	Patient sedated with Midazolam and Ultiva. Connected to the MAV in PCV. ABG: Ph 7.43, 34 PCO ₂ , PO ₂ 92, HCO ₃ - 22.6, -1.7 Well, SaO ₂ 97%. Hemodynamic supported by Norepinephrine 12:10 µg / kg / min. FC 59 bpm, PA 120/79, SPO ₂ 100%. Urine output 1400ml / 24h. BE / U +1500. Afebrile
----------	------	---

Figure 4.2: Informazioni non strutturata

Exam	Measured value	Measurement unit	Target range
Glycemia	360*	mg/dl	60 – 100 (eseguita su siero)
Azotemia	30	mg/dl	10 - 50
Creatinine	0.92	mg/dl	0.50 – 1.10
Sodium in Serum	139	mEq/L	135 – 147
Serum potassium	3.8	mEq/L	3.5 – 5.1
Chlorine in Serum	102	mEq/L	98 - 108

Figure 4.3: Informazioni strutturali

4.1.2 Fase di estrazione e analisi delle informazioni

Conclusa la fase precedente, si passa alla fase di estrazione e analisi dei dati della cartella clinica in formato digitale. Per fare ciò, lo sviluppatore ha istituito due classi Java che permettono l'estrazione dei dati in entrambi i formati concessi dal tool (.docx e .pdf), e sono:

- **DocxReader.java**
- **PDFReader.java**

Nella classe *DocxReader* si utilizza la classe *XWPFWordExtractor*[19] di *ApachePOI*, mentre in *PDFReader* si utilizza la classe *PDFTextStripper*[20] di *PDFBox*. Entrambe le classi forniscono come risultato un "formatted raw text" che viene definito come "testo grezzo" poichè c'è la necessità di mantenere tutte le caratteristiche principali del testo, come: eventuali spazi tra le parole, ritorno a capo, tab, ecc, che è possibile tenere in considerazione per riconoscere la struttura delle informazioni e a cui è possibile fare riferimento attraverso le espressioni regolari. Finita la fase di estrazione, si passa alla fase di analisi di tali dati, per ricavarne informazioni. In un primo momento si può pensare che i dati vengano estratti e successivamente analizzati in maniera simultanea. Chiaramente, non è così: dopo l'estrazione dei dati che chiameremo *documento*, viene suddiviso in parti più piccole dette *sentence*, e vengono analizzati una per volta.

L'artefice dell'analisi dei dati, viene effettuata dalla classe *NLP.java*, che sfruttando la libreria *Stanford CoreNLP* e alcune espressioni regolari, "tentando di capire" il tipo di informazione che ha ricevuto in input. Bisogna tener conto dei tipi di informazioni che la classe riesce ad

analizzare, che sono:

- **Pattern noti:** ovvero valori di tipo "chiave: valore. Per capire il significato del pattern (ad esempio, se si tratta di una *sentence* che contiene informazioni personali sul paziente e/o informazioni sul ricovero, ecc.) è stato usato un *ner rule-based* (un file .txt denominato *Entity*) che viene mappato al *regexner* di *Stanford CoreNLP*. Per altri tipi di informazioni, si utilizza anche *MetaMap Lite*.
- **Diario Clinico:** rappresentano le informazioni che sono contenute all'interno del diario clinico, quindi *non strutturate* su cui vengono effettuate le analisi *NLP* per il riconoscimento di *parametri di laboratorio, terapie, sintomi*. La *sentence* in input all'algoritmo *NLP* viene suddivisa in varie parti, ognuna delle quali è processata sia utilizzando *Stanford CoreNLP* che *MetaMap Lite* (per riconoscere concetti *UMLS*).

L'analisi dei concetti biometrici tramite *MetaMapLite*, è delegata ad un'altra componente (*MetaMapMatcher*), la quale mappa ogni termine medico rilevato ad una o più categorie semantiche. Chiaramente, *MetaMapLite* trova molti più concetti di quelli richiesti (terapie, sintomi e parametri di laboratorio), quindi alcuni di questi non saranno presi in considerazione. In più, vengono rilevati **acronimi** o **abbreviazioni** attraverso un'espressione regolare. Attraverso poi, *depparse* di *Stanford CoreNLP*, si cerca di recuperare ulteriori informazioni relative ai dosaggi o ad eventuali note per le terapie. La *Figura 4.4* schematizza il processo di analisi del diario clinico attraverso un esempio.

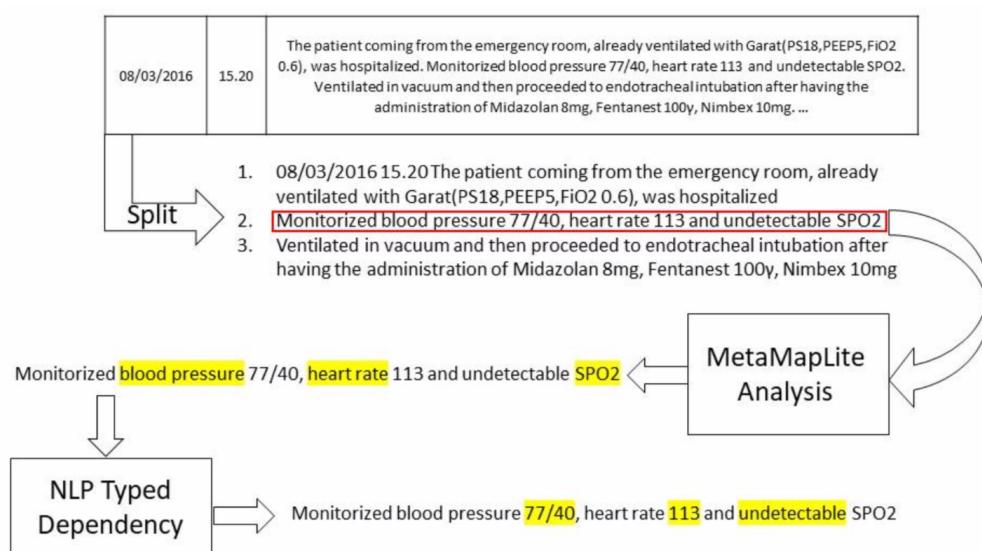


Figura 4.4: Esempio di analisi del diario clinico

I passi da seguire sono:

1. Estrazione del "formatted raw text" dal diario clinico viene estratto il testo.
2. Il testo estratto viene suddiviso in *sentence* grazie all'uso di espressioni regolari che, rilevando un "."(punto) capiscono che la *sentence* finisce ed è pronta per essere analizzata tramite *Stanford CoreNLP*. La figura mostra un rettangolo rosso che possiamo prendere come riferimento per capire al meglio quanto appena spiegato.
3. La *sentence* di riferimento è analizzata anche tramite *MetaMapLite*, che consente di individuare i concetti *UMLS* che sono successivamente evidenziati in giallo nella figura.
4. A partire dai concetti individuati, si analizzano le *typed dependencies* generate dall'analisi di *Stanford CoreNLP* in cui tali concetti sono coinvolti per individuare le informazioni ad esso collegati.

Tutte le informazioni estratte vanno a riempire le strutture di alcune classi predefinite, ognuna delle quali contiene informazioni corrispondenti a ciò che rappresenta (ad esempio, c'è una classe *Patient* che contiene informazioni del paziente, ecc.)

4.1.3 Creazione e validazione del file XML

Successivamente alla reperibilità dei dati letti dalla cartella clinica, viene generato un file XML dentro il quale sono rappresentate le informazioni ottenute circa la fase di estrazione. La classe `XMLWriter` si occupa della creazione e della modellazione di tale file. Per esempio, un parametro di laboratorio viene modellato tramite il metodo in Figura 4.5: il metodo riceve in input un'istanza di `Parameter` (classe rappresentante un parametro) e il nodo padre, a cui collegare il nuovo nodo (parametro) che sarà creato e, di conseguenza, i suoi figli (descrizione, valore misurato, unità di misura, range di riferimento, data e ora). La creazione del file `.xml`

```
private void setParameter(Parameter p, Element father) {
    Element parameter = this.doc.createElement("parameter");
    Element description = this.doc.createElement("description");
    description.appendChild(this.doc.createTextNode(p.getDescription()));
    parameter.appendChild(description);

    Element measuredValue = this.doc.createElement("measuredValue");
    measuredValue.appendChild(this.doc.createTextNode(p.getMeasuredValue()));
    parameter.appendChild(measuredValue);

    Element measurementUnit = this.doc.createElement("measurementUnit");
    measurementUnit.appendChild(this.doc.createTextNode(p.getMeasurementUnit()));
    parameter.appendChild(measurementUnit);

    Element targetRange = this.doc.createElement("targetRange");
    targetRange.appendChild(this.doc.createTextNode(p.getTargetRange()));
    parameter.appendChild(targetRange);

    Element dateTime = this.doc.createElement("dateTime");
    dateTime.appendChild(this.doc.createTextNode(p.getDateTime()));
    parameter.appendChild(dateTime);

    father.appendChild(parameter);
}
```

Figure 4.5: Modellazione di un parametro in XML

successivamente, questo viene sottoposto a validazione tramite la componente `XSDValidator`, che attraverso il metodo `validateXMLSchema` il quale prevede due parametri in input: `xsdPath` e `xmlPath`, secondo cui uno è il file XML da validare, e l'altro è il file XSD che si ha da "linea guida" per validare il primo. Una volta che la validazione abbia esito positivo, viene schematizzato secondo la seguente figura:

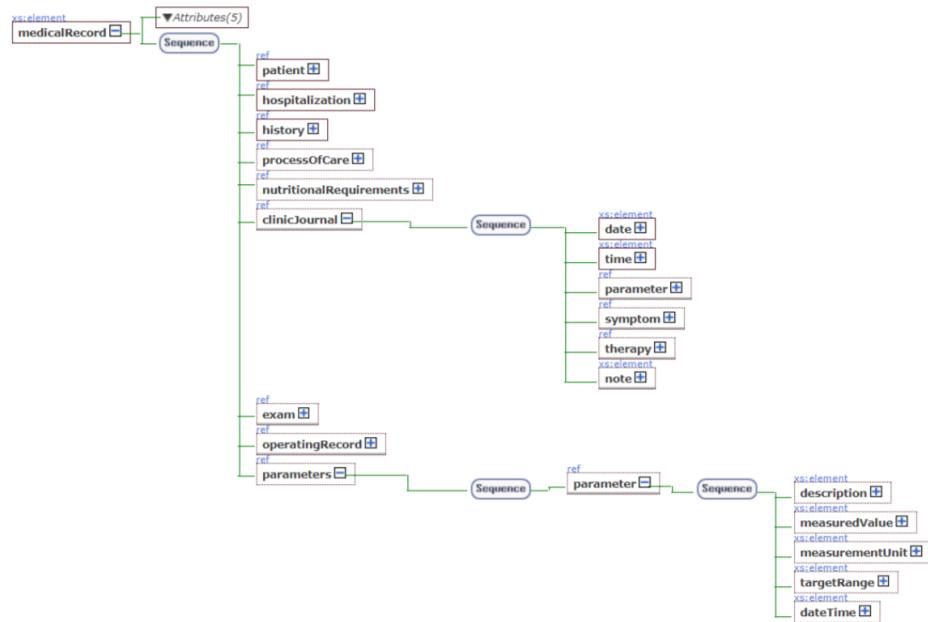


Figure 4.6: Estratto visuale del file XSD utilizzato per la validazione

Se il file XML supera la fase di validazione, allora si passa alla fase successiva, ovvero il popolamento dell'ontologia.

4.1.4 Popolamento ontologia

La classe adottata per il popolamento dell'ontologia è: *OntologyWriter.java*, che sfrutta i metodi messi a disposizione da *Apache Jena*, che si occupa dell'aggiornamento dell'ontologia. Si parla di aggiornamento poichè una volta analizzata la prima cartella clinica, le successive, non prenderanno il posto della prima, ovvero, i dati scritti in precedenza non saranno cancellati, bensì si andranno ad aggiungere ai dati già presenti, il che contribuirà solo ad aumentare la base di conoscenza ontologica. Si specifica che un meccanismo di questo tipo porterà ad ottenere un'ontologia di dimensioni non trascurabili, e a contenere molti (se non tutti) dei concetti medici attualmente conosciuti. A valle di quanto appena detto, il capitolo successivo spiegherà meglio le problematiche che si sono riscontrate e come sono state gestite.

4.1.5 Fase di inferenza

L'ultima fase del processo di diagnosi automatica riguarda l'individuazione di nuovi concetti a partire da quelli noti. Tale processo è detto *inferenziale*. *Apache Jena* permette di effettuare il processo di inferenza attraverso i *reasoner*.

Tale fase ha visto, quindi, la creazione di regole logiche, in un file *.txt*, per la derivazione di alcuni indicatori parametrici. Tale classe prevede la lettura del modello ontologico costituito fino a quel momento e delle regole logiche da applicare a tale modello al fine d'individuare nuovi fatti che, al termine del processo di inferenza, saranno aggiunti al modello ontologico di partenza. I reasoner si possono personalizzare sulla base del tipo di informazioni che si vogliono inferire.

4.1.6 Presentazione del tool

Il sistema è corredato da una semplice interfaccia grafica, classe *Design.java*, che permette il caricamento di una cartella clinica in *.docx* o *.pdf*, ai fini di effettuare l'analisi, e l'esecuzione del processo inferenziale sull'ontologia creata.

Il processo essendo corposo, può richiedere alcuni minuti per essere completato. La *Figura 4.7* e *Figura 4.8* mostrano l'utilizzo del tool.

In particolare, in *Figura 3.11* si possono vedere i passaggi da effettuare per creare l'ontologia, a partire da un file in formato *.docx* e *.pdf*. Come si evince dalla figura, basta cliccare su *File>Upload medical record*, scegliere il formato che più ci soddisfa e cliccare *Upload*. A questo punto, si dovrà attendere il completamento del processo di creazione dell'ontologia, al termine del quale verrà visualizzato un messaggio per informare l'utente che può eseguire il processo d'inferenza.

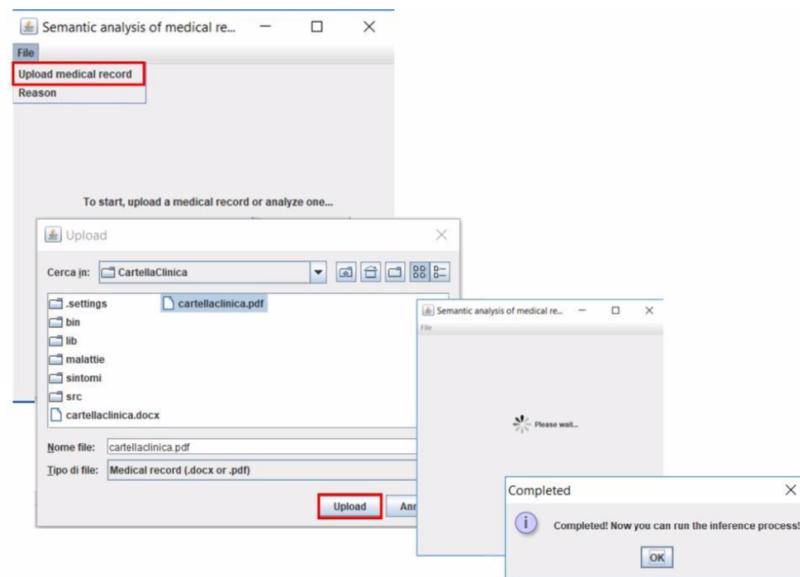


Figura 4.7: Processo grafico per l'analisi della cartella clinica in formato digitale

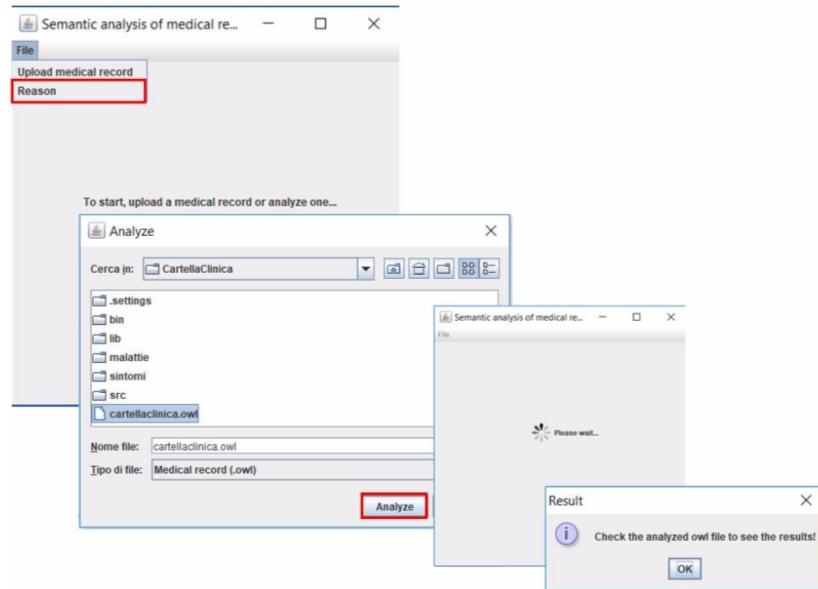


Figura 4.8: Processo grafico per eseguire il processo inferenziale

In Figura 4.8, invece, si possono vedere i passaggi per eseguire il processo inferenziale sull’ontologia creata. A tal scopo, basta cliccare su *File>Reason*, scegliere il file ontologico da analizzare (avente formato *.owl*) e cliccare su *Analyze*. A questo punto si dovrà attendere il completamento del processo di inferenza, al termine del quale verrà visualizzato un messaggio per informare l’utente di visualizzare l’ontologia (tramite l’applicativo **Protégé**) per vedere i concetti inferiti.

4.1.7 L’installazione del tool e le sue problematiche

Per poter utilizzare il tool, oltre a studiare a fondo il pieno funzionamento delle varie componenti che lo concepiscono, bisogna interfacciarsi con determinati tipi di problematiche che impediscono l’installazione.

Il tool ha un livello di complessità media/alta, dovuto all’elevato numero di librerie esterne che esso utilizza per effettuare l’analisi e l’estrazione dei dati che sono state descritte in precedenza. La Figura 4.9 mostra solo alcune delle 64 librerie esterne utilizzate. Uno dei primi problemi, lo si riscontra quando si tenta di importare il progetto su un’altra macchina, poichè le librerie esterne non sono rese permanenti al progetto. E’ stato dunque, necessario scaricare ed importare manualmente tutte le librerie e successivamente creare una cartella *lib* interna al progetto in modo tale che siano sempre memorizzate nel file *.classpath* del progetto, ovviando così il problema di non avere le librerie necessarie.

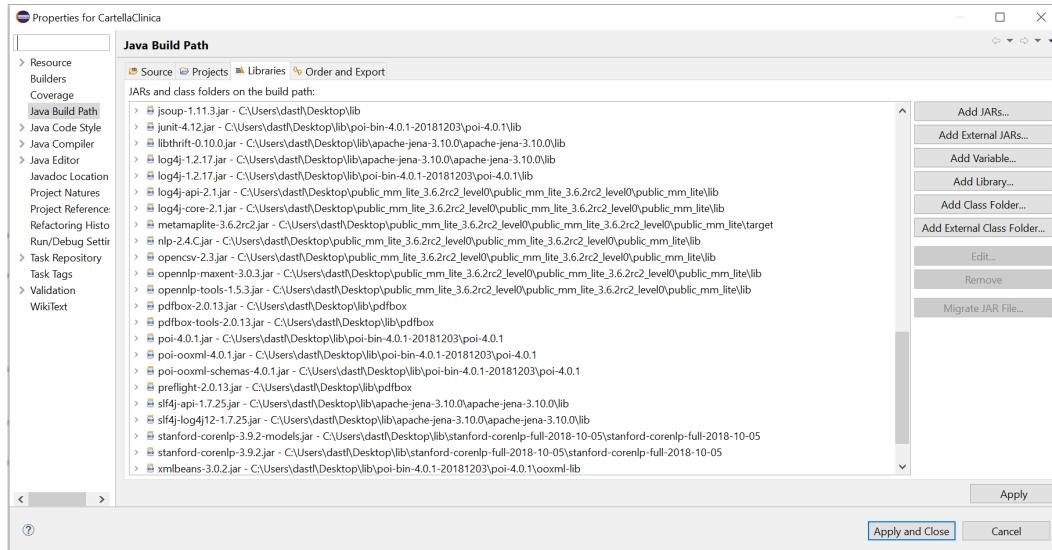


Figure 4.9: Una parte delle librerie esterne utilizzate

La seconda problematica riscontrata, è relativa ai *path* delle cartelle cliniche da far analizzare al tool. La *Figura 4.10* mostra (nel riquadro rosso) l'istruzione che prende in input un *path* di una cartella clinica (in formato *.pdf*) in modo non automatico, bensì statico e quindi non modificabile al momento dell'esecuzione del tool.

```
public void run() {
    try {
        startTime = System.currentTimeMillis(); //mi prendo il tempo di inizio
        System.out.println(file.getAbsolutePath());
        if(FilenameUtils.getExtension(file.getAbsolutePath()).equals("docx"))
            new DocxReader(file.getAbsolutePath());
        else if(FilenameUtils.getExtension(file.getAbsolutePath()).equals("pdf"))
            //RIGHT
            new PDFReader(file.getAbsolutePath());
        //new PDFReader("/Users/UniSa/Desktop/TESI/Progetto cartella clinica/Progetto cartella clinica/cartellaclinica.pdf");
    } catch (Exception e) {
        throw new Exception();
    }
}
```

Figure 4.10: Path statico

La modifica da apportare consiste nel far in modo che il parametro che la classe *PDFReader* richiede in input, sia trovato in modo automatico dal tool al momento dell'esecuzione, utilizzando il metodo *getAbsolutePath()*. Il riquadro verde, mostra quanto appena spiegato.

Come già è stato detto in precedenza, il software, data la sua complessità, richiede anche caratteristiche hardware importanti.

Nel momento in cui il software parte con la sua esecuzione, può presentare un errore di "Java Heap Space", il che significa che la JVM (*Java Virtual Machine*) non dispone della potenza di calcolo sufficiente a poter portare a termine l'esecuzione del programma, bloccandosi

in modo inaspettato e terminando la sua esecuzione. Tuttavia, la maggior parte degli IDE (*Integrated Development Environment*) consentono di modificare i parametri alla JVM in modo da dedicare la potenza di calcolo necessaria per portare a termine l'esecuzione. La *Figura 4.10* mostra come i parametri della JVM possano essere modificati. I parametri possono essere di due tipi:

- **-Xmx<size>**: per specificare la massima dimensione dell'heap
- **-Xms<size>**: per poter specificare la dimensione iniziale dell'heap Java
- **-Xss<size>**: per poter impostare la dimensione dello stack del Thread Java

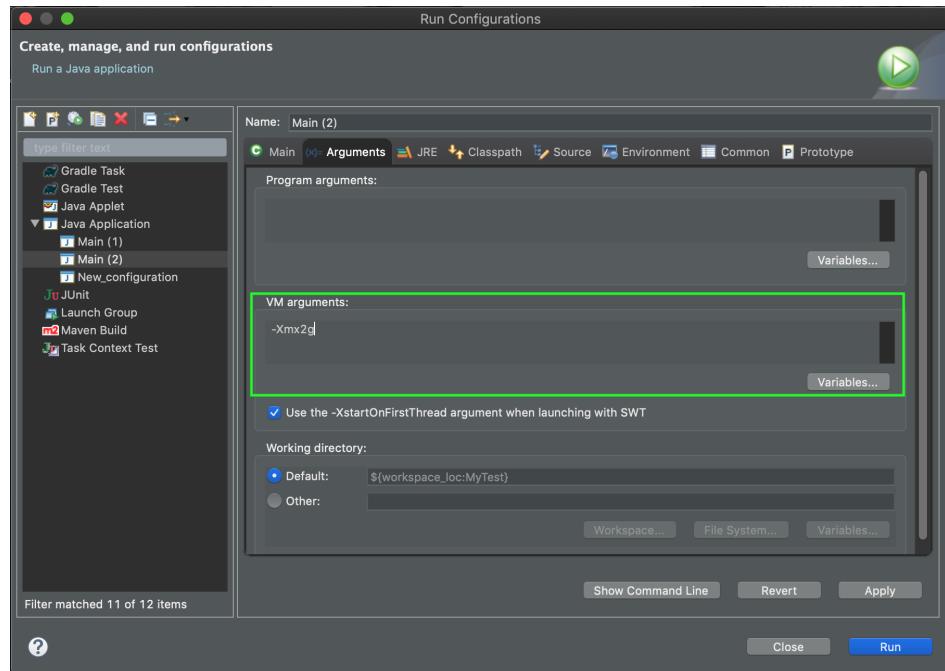


Figure 4.11: Parametri per la VM

Si precisa che la "size" può essere espressa in termini di **Gigabyte** o **Megabyte**. Il problema però, potrebbe persistere, motivo per cui, si consiglia di installare il software su una macchina più potente.

CHAPTER 5

Valutazione Sperimentale

In questo capitolo, saranno esposti il processo di sperimentazione effettuato e i risultati ottenuti, elencando caratteristiche e informazioni relativi ai pazienti per diagnosticare la *SIRS*. Innanzitutto, verrà presentata una breve panoramica sulla *SEPSI*

5.1 La Sepsi

La *SEPSI* [16] [26] [9] [18] è una sindrome clinica caratterizzata da un'abnorme *Risposta Infiammatoria Sistemica (SIRS)* messa in atto dall'organismo in seguito a un'infezione *Figura 5.1*. Colpisce circa 700.000 persone ogni anno solo in Europa e un caso su cinque è fatale. Negli Stati Uniti, invece, i nuovi casi di Sepsi sono stimati essere 750.000 ogni anno, con un'incidenza che è probabilmente destinata ad aumentare dell'1,5% all'anno, a causa dell'invecchiamento della popolazione. Vista la sua natura particolarmente aggressiva e multifattoriale, la *Sepsi* conduce rapidamente a morte e costituisce la principale causa di decesso nelle terapie intensive non coronarie di tutto il mondo, con tassi di letalità che vanno dal 20% per la *Sepsi*, al 40% per la *Sepsi grave*, ad oltre il 60% per lo *shock settico*: cumulativamente, nel mondo, muoiono, per *Sepsi*, circa 1.400 persone al giorno.

Si parla di *Sepsi grave* quando alla *Sepsi* è associata almeno una disfunzione d'organo lontano dalla sede di infezione, ipotensione o ipoperfusione. Lo *shock settico*, invece, è rappresentato da una *Sepsi grave* caratterizzata da un'ipotensione arteriosa non riconducibile ad altre cause e che non risponde alla riespansione volemica [2].

La *Sepsi* si presenta, quindi, quando il sistema immunitario perde il controllo e inizia a danneggiare gli organi e i tessuti. Tutto inizia con un'infezione locale. I germi responsabili dell'infezione entrano nel circolo sanguigno e il sistema immunitario reagisce in modo anomalo. L'alterata riposta immunitaria all'infezione causa un'infiammazione generalizzata e può portare all'insufficienza di più organi e, infine, al decesso. Le possibilità di sconfiggerla, quindi, sono tanto maggiori quanto prima viene riconosciuta e trattata. La diagnosi di Sepsi,

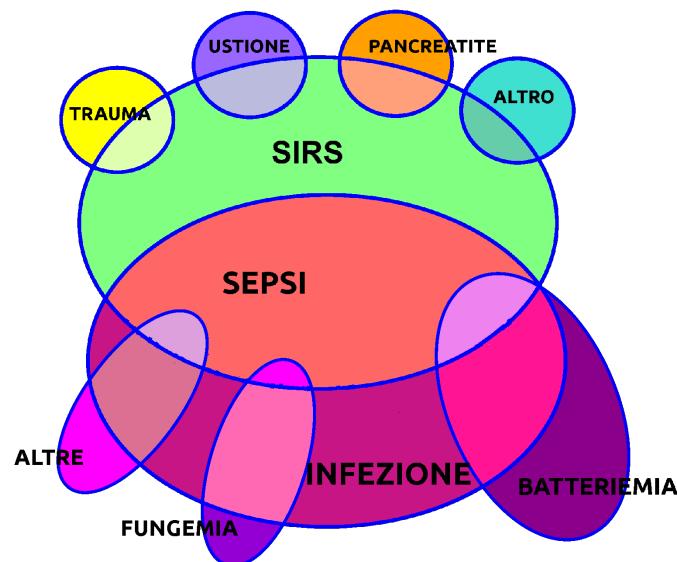


Figure 5.1: Relazione tra Sirs, Sepsi ed Infezione

accettata fino all'inizio del 2016, veniva posta in seguito al riscontro di almeno due dei criteri che identificano la *SIRS*, accompagnati da un'infezione. I criteri per definire la *SIRS* furono concordati nel 1992. La *SIRS* può essere diagnosticata quando sono presenti **almeno due** delle seguenti condizioni:

- **Frequenza cardiaca** superiore a 90 battiti al minuto;
- **Temperatura corporea** inferiore a 36°C o stato febbrile (>38°fino a 41°);
- **Frequenza respiratoria** > 20 atti/minuto (*tachipnea*) o pressione
- **Numero di globuli bianchi** nel sangue inferiore ai 4.000 per mm³ (*leucopenia*) o superiore ai 12.000 per mm³ (*leucocitosi*), oppure aumento superiore al 10% di forme immature di neutrofili (*Large Unstained Cells*)

L'utilizzo del concetto di *SIRS* per la definizione di *Sepsi* è gravato dal problema della scarsa specificità, in quanto molteplici condizioni cliniche non infettive possono essere associate ad

un quadro di *SIRS*. Nell’ambito del Pronto Soccorso moltissimi pazienti presentano criteri compatibili con una *SIRS*, ma solo alcuni presentano un’infezione associata; inoltre, non sempre un’infezione determina una risposta infiammatoria.

5.2 La sperimentazione del tool

Dopo aver studiato la struttura del software, immedesimandosi in ogni singola componente, capendone tutte le funzionalità, è stato necessario studiare - come detto anche nei capitoli precedenti- la struttura delle cartelle cliniche dei vari pazienti.

La prima fase consiste nella lettura e comprensione del diario clinico, quanto più minuziosa possibile, ricordando che ogni singola informazioni potrebbe essere rilevante. Successivamente, si è passati alla trascrizione dei dati in un formato digitale (sono stati considerati i documenti *.docx* poichè direttamente a contatto con i medici e infermieri). Nonostante l’elevata attenzione durante questa fase, alcuni dati sono stati rilevati come illegibili, causa l’ambiguità della grafia e, la velocità con cui sono stati scritti. Successivamente discuteremo del calcolo di dati illegibili per ogni cartella clinica, ricordando che, seppur illegibili, hanno un alto tasso di probabilità di contenere dati che soddisfano i criteri di *Sirs* sopra descritti. La *Figura 5.2* mostra un estratto del diario clinico, mostrando l’incomprensibilità dei dati e, successivamente la *Figura 5.3* mostra come i dati non compresi, siano stati tradotti all’interno del modello di cartella clinica digitale.

DATA	DIARIO CLINICO	TERAPIA
8/11/16	<p>ave 1500 g. b. intollerante al pane diabetico con regi. di edema periferico di nausea. Ibrido diffuse labiale per irritabile costale dolori testicolari, male per recto retto Ano TAC eseguito in radiologia nella clinica della TAC nello stesso giorno prescritto generale in clinica D. C. P.G.D.S. in oggi.</p> <p style="text-align: right;"><i>P. G. D. S.</i></p>	

8/11/16
 13,35 Comparsa riacquisto ematoma clinico conente
 anemia gassino ciechi diffuse, in terapia
 con maschera facciale; PA 75/40 FC 100b/m.
 Sol. O₂ 65%. Si procede col immediato I.v. e
 si anestetizza con Anabut e O₂ 100%; si fa
 metacolo sternone TC linea e addome (valori
 e corrispondenti chiamate insieme in corso.
 E.G.A. p.ti Y.12 p.cq. 86.0 p.o. 49.5 Hb 15.5 SG 71.9
 BE -0.1 HCO₃ 20.8. Pidice Volumen 500 ml +
 Fisiologico 500 ml; inizio infusione di Ringer lactato 1000 ml.
 Vista l'estremo gravità delle condizioni
 cliniche, imposta la cardiochirurgia
 si opera prima in Ringer lactato.

Figure 5.2: Diario clinico illegibile

Affinchè l'elaborazione dei dati possa ottenere esito positivo, è necessario trascriverli secondo un pattern stabilito dal developer del tool che rispetta quelle che sono le indicazioni di rappresentabilità della cartella clinica. L'estrazione dei dati viene effettuata utilizzando la classe *DocxReader.java* e *PDFReader.java* (descritte al Capitolo 3). Successiva all'estrazione, viene l'analisi, che, tramite la classe *NLP.java* sfrutta le librerie di *Stanford CoreNLP*, tentando di "capire" il tipo di informazione che sta estrapolando. La Figura 5.4 mostra l'interfaccia grafica che permette di scegliere le cartelle cliniche a secoda del formato preferito.

Daily Clinic Journal		
SURNAME:	NAME:	MEDICAL RECORD NUMBER:
DATE	TIME	
	13.30	The patient performed urgent abdomen CT and alerted the emergency surgeon for transfer to their ward. REANINATORE is also alerted for transfer to their ward due to the very serious deterioration of the general clinical conditions. EGA is practiced in ventimask with FiO2 50%
	14.00	UNREADABLE
07/11/2014	13.35	Resuscitative advice. Patient in very serious clinical condition, absent consciousness, gasping, widespread cyanosis, in O2 facial mask therapy. ABP 75/40 HR 100 bpm SatO2 65%. We proceed to immediate IOT and assist with Ambu and O2 100%. This morning he has already practiced chest and abdomen CT and ongoing urgent surgical counseling. EGA pH 7,142 PCO2 86.0 pO2 49.5 Hb 15.5 SO2 77.9 BE -0.1 HCO3- 20.8. Practice voluven 500 ml with physiological 1000 ml. Begins norepinephrine infusion in P/S given the extreme severity of clinical conditions, respiratory and cardio-circulatory. He goes to resuscitation.
	14.10	Resuscitation. The patient enters CR from Bronchoncology. He practiced abdomen CT with a report of "acute pancreatitis in the hemorrhagic necrotic phase (stage IV according to Balthazar)". Serious clinical conditions. Intubate, ventilated, cologned at RAM and axed in SIMV with FiO2 0.8. It monitors vital parameters, practice laboratory tests, ECG, EGA. Ongoing norepinephrine infusion at high dosages. ABP 75/40 HR 114 bpm T 36 SatO2 90% diuresis at 500ml input. EGA, pH 7,138, PCO2 80.5, PO2 67.5, Hb 15.3, SO2 91.9, K-2.2, Na-133, Glu 115, BE -20, HCO3- 19.9.
	14.20	Bradycardia with arrhythmia. Practice atropine 1f plus 1f with heart rhythm resumption. Continuous infusion of norepinephrine in P/S. ABP 75/40 HR 96 bpm SatO2 94%. Practice voluven 500 ml with physiological 500 ml. Waiting for laboratory tests practice the following therapies: physiological 3000 ml for 24, Antra 1f for 12, Urbason 40 mg for 12. Aerosol (clenil and Ventolin) for 8, Foy 6 fl in P/S for 24. Lasix 1f for 12, 2 fl kCl in P/S, 50 physiological for 24, Midazolam in P/S -> 5 ml /h, Clexane 6000 for 12, Albumin 2 fl for 8.
	16.00	It requires advice for NPT and surgical control. EGA monitoring. According to laboratory data you increase the KCl. ABP 120/70 HR 78 bpm, SatO2 97%. Diuresis from the 400ml entrance

Figure 5.3: Gestione dati illegibili

Dopo aver estrapolato tutti i dati, il tool, genera un file XML che mostra le informazioni estratte dalla cartella clinica di un paziente e, successivamente ne esegue la validazione, controllando che il file XML generato ripetti le linee guida del file XSD già presente all'interno della struttura del software. Se l'esito della validazione risulta essere positivo, allora la prima esecuzione del tool può considerarsi andata a buon fine, consentendo di interagire nuovamente con l'interfaccia grafica del tool.

Si sottolinea che nonostante l'esito positivo della prima fase di esecuzione della cartella clinica, non implica che il paziente in questione riscontri criteri di **Sepsi**. Nella *Figura 5.5* il tool ci notifica che il processo è stato concluso e nella *Figura 5.6* è mostrato il file XML validato, contenente i dati estrapolati dalla cartella clinica.

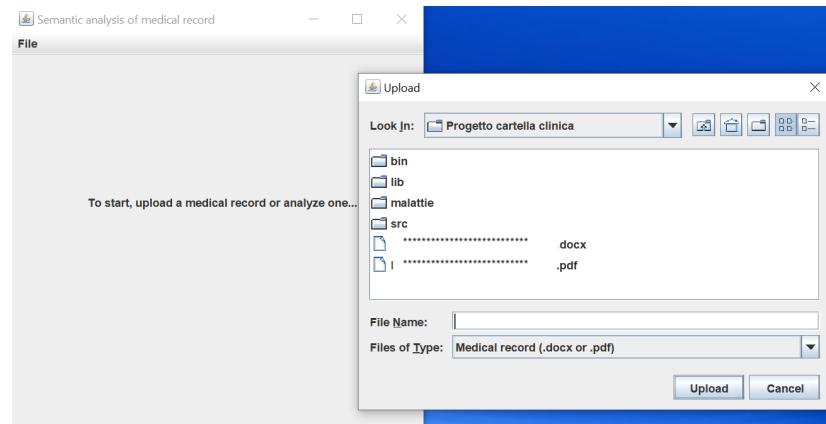


Figure 5.4: Scelta formato della cartella clinica

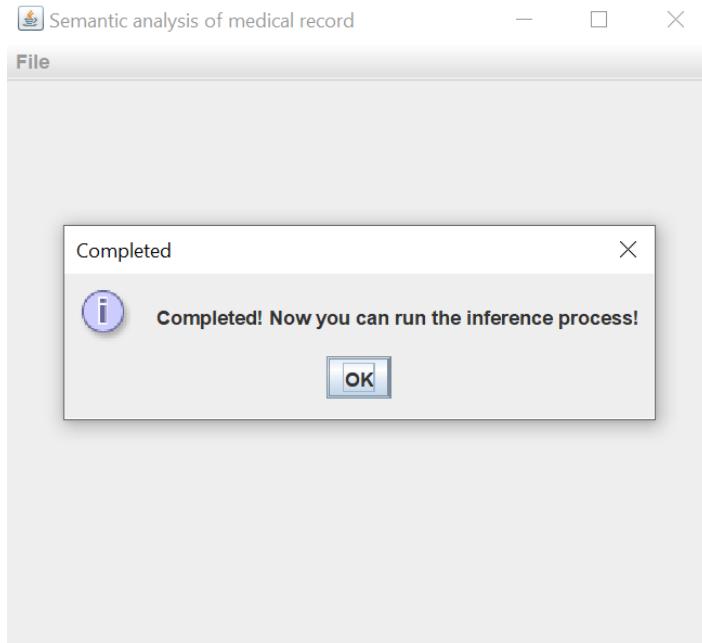
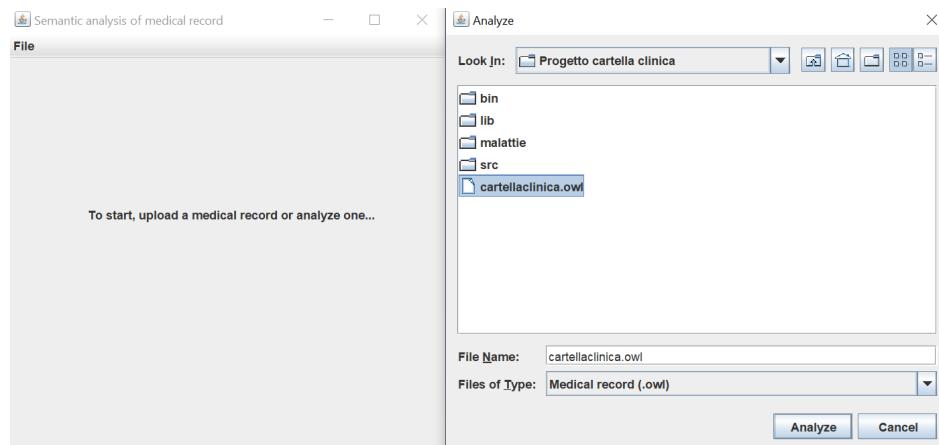


Figure 5.5: Il processo è stato completato con successo

Dopo aver effettuato l'estrazione dei dati, tramite la classe *OntologyWriter.java* si sfruttano i metodi messi a disposizione da *Apache Jena* atti ad ottenere l'aggiornamento dell'ontologia. Come già detto nei capitoli precedenti, l'aggiornamento, porterà ad ottenere un'ontologia di dimensioni non trascurabili. Motivo per cui durante la fase di sperimentazione, ci si accorge che l'ontologia può contenere al più tre rilevazioni di tre pazienti diversi, poichè nel migliore dei casi il tempo di esecuzione risulterà essere molto grande. Nel peggiore, invece, il software darà errore sul tipo di dati letti, cosa che non succede se si utilizza un'ontologia vuota. La *Figura 5.7* mostra l'esecuzione del processo inferenziale sull'ontologia creata, mentre la *Figura 5.8* mostra il messaggio di completamento.

Node	Content
xml	version="1.0" encoding="UTF-8" standalone="no"
medicalRecord	http://www.w3.org/2001/XMLSchema-instance
@ xmlns xsi="http://www.w3.org/2001/XMLSchema-instance"	10104031010 – Resuscitation – P.O. 01
@ approvalUnitCode	
@ hospital	
@ hospitalCode	
@ hospitalisationYear	2014
@ number	2014042680
@ xsd: schemaLocation	/schemainglese.xsd
patient	***** *****
@ name	M
@ surname	20/10/1943
@ gender	CAVA DE TIRRENI
@ dateOfBirth	71
@ placeOfBirth	*****
@ age	Italian
@ fiscalCode	-
@ nationality	333/3418685
@ contactPerson	
@ phone	
@ residence	
hospitalisation	09/11/2014
@ hospitalisationDate	
@ acceptanceDiagnosis	
@ finalAssessment	
@ dischargeSummary	
history	Summary medical history: parkinson's disease, Chronic brain vacuopathy, Right shoulder c disease medical history
@ history	
@ disease	
@ disease	

Figure 5.6: Struttura del file XML con i dati estrapolati**Figure 5.7:** Esecuzione del processo inferenziale

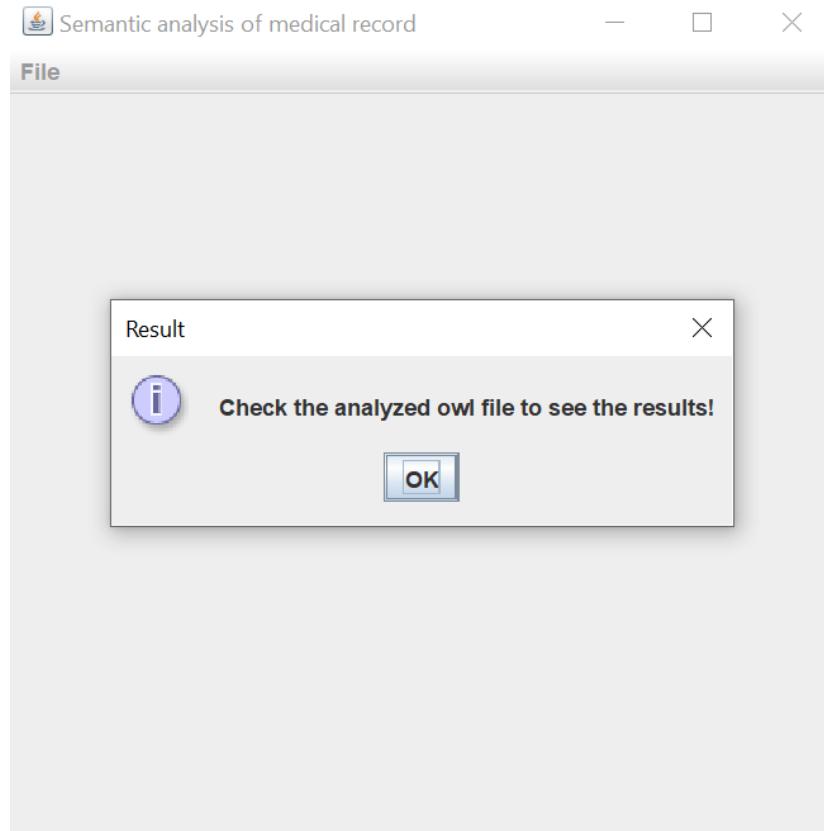


Figure 5.8: Messaggio di completamento

Se l'esito risulta essere positivo come la figura soprastante, allora si posso vedere i risultato aprendo il file *cartellaclinica.owl* tramite l'applicativo **Protégé** progettato e distribuito dal centro di ricerca informatica biomedica dell'Univerità di Stanford [25].

Riassumendo il processo di sperimentazione effettuato per la valutazione del risultato di analisi di ogni cartella clinica presa in esame, ha previsto l'applicazione dei seguenti passi:

1. **Comprensione** della cartella clinica nel suo formato originale, cercando (per quanto possibile) di ottenere tutti i dati clinici, incluse le terapie.
2. **Trascrizione** della cartella clinica avendo cura di tradurre tutti i passi clinici in inglese utilizzando il formato **.docx** e successivamente ottenere il file **.pdf**. Il tutto seguendo il *pattern* mostrato anche in *Figura 4.3*.
3. **Avvio software e selezione cartella clinica** e validazione dell'esito.
4. **Esecuzione processo inferenziale** in modo da permettere la visualizzazione dei dati estratti.

5.2.1 Breve introduzione all’applicativo Protégé

Protégé è un editor ontologico gratuito e open source e un sistema di gestione della conoscenza. Protégé fornisce un’interfaccia utente grafica per definire le ontologie. Include anche classificatori deduttivi per convalidare la coerenza dei modelli e inferire nuove informazioni basate sull’analisi di un’ontologia. Questa applicazione è scritta in Java e utilizza fortemente Swing per creare l’interfaccia utente. Protégé ha recentemente oltre 300.000 utenti registrati. La *Figura 5.9* mostra l’interfaccia grafica dell’applicativo.

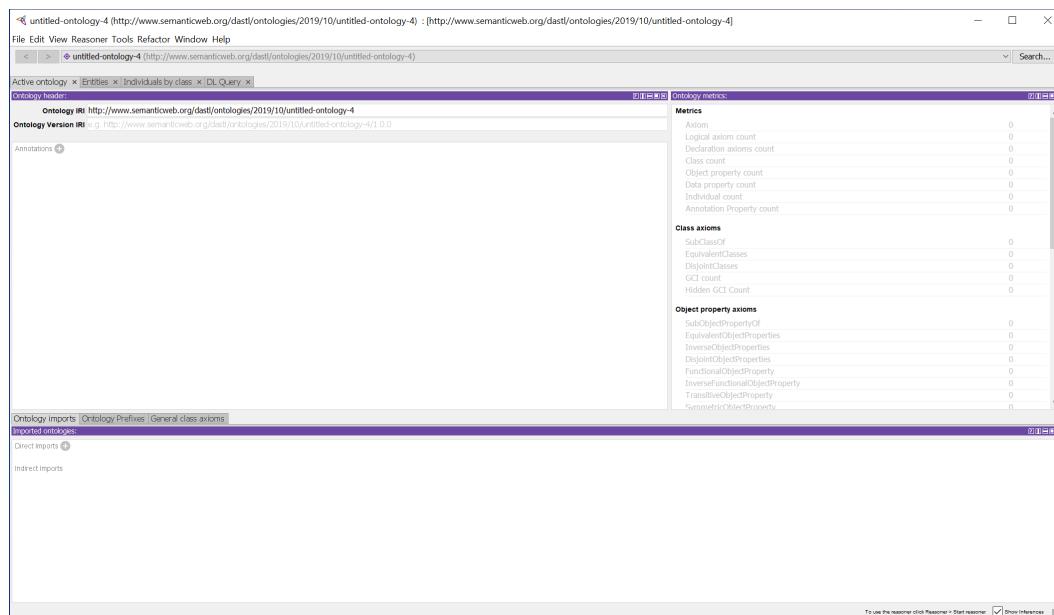


Figura 5.9: Interfaccia grafica di Protégé

5.3 Risultati ottenuti dall’elaborazione del software

In questa sezione vengono esposti gli esiti relativi alle cartelle cliniche prese in considerazione. Grazie all’utilizzo dell’applicativo sopra citato è stato possibile revisionare gli output restituiti dal tool dopo aver analizzato e inserito singolarmente ogni cartella clinica. Su una base di circa sette cartelle cliniche, provenienti dall’Ospedale di Salerno, per ogni paziente, si è reso necessario calcolare le seguenti metriche:

- **Tempi di esecuzione per PDF o docx:** Durante l’analisi e l’estrazione dei dati (sia per il formato .pdf che .docx) si calcolano i tempi di esecuzione del tool, stabiliti in secondi ;
- **Tempi di esecuzione per il file OWL:** Il calcolo coincide con la seconda esecuzione del software, ossia l’esecuzione del file di inferenza, espressi sempre in secondi;

- **Totale "UNREADABLE"** presenti all’interno di un *medical record*: ottenuti durante la trascrizione della cartella clinica nel formato gititale;
- **Numero dei giorni di ricovero;**
- **Numero dei giorni in cui si presenta la SIRS;**
- **Conteggio dei caratteri nel file .docx;**

Ognuna di queste metriche è volta ad ottenere indagini statistiche sulle elaborazione del tool, cercando dipendenza che possano fornire informazioni.

Le statistiche sulle cartelle cliniche dei pazienti, con le metriche precedentemente spiegate, sono riportate nelle tabelle seguenti, che per motivi di privacy non riporteranno i nomi dei pazient:

Table 5.1: Metriche relative ai pazienti

Paziente	T.esecuzione PDF	UNREAD	Ricovero	Esami clinici	N°caratteri
Paz.1	20 s	3	46	9	26.235
Paz.2	26 s	0	17	8	26.400
Paz.3	39 s	34	38	10	31.877
Paz.4	36 s	7	36	9	27.231
Paz.5	35 s	2	32	9	23.896
Paz.6	30 s	4	39	7	19.541
Paz.7	26 s	4	15	12	14.561

Table 5.2: Metriche aggiuntive

Paziente	T.esecuzione OWL	N°giorni SIRS
Paz.1	66 s	9
Paz.2	18 s	3
Paz.3	213 s	4
Paz.4	128 s	10
Paz.5	107 s	4
Paz.6	38 s	2
Paz.7	25 s	2

Alla luce dei dati esaminati, e delle metriche riportate, si è giunti alla conclusione che tutti i pazienti esaminati hanno riportato casi di *Sirs* in molteplici giorni e, il tool li ha riscontrati in maniera eccelsa. Sebbene sia un esito positivo, che riporta il buon funzionamento del software, si precisa che il file d’inferenza non è rimasto lo stesso, ovvero: ad ogni esame-nazione dei *medical record* è stato necessario “re-inizializzare” il file *.owl*, causa gli elevati tempi di esecuzione che hanno raggiunto picchi di 40 minuti circa e per errori di struttura dei file *.xml* e la relativa validazione. Nelle sezioni successive verrano fornite ulteriori controprove che dimostrano la veridicità dell’algoritmo.

Nella tabella seguente vengono riportate le medie relative ai tempi di esecuzione per l’analisi dei file (*.pdf* o *.docx*) e dei file *owl*:

Table 5.3: Medie dei tempi di esecuzione

Media tempi esecuzione PDF o docx	Media tempi esecuzione OWL
30,28 s	85 s

Con l’aiuto dei precedenti valori ottenuti testando il tool, si forniscono i grafici e le relative spiegazioni associate ad essi. Il primo caso ad essere trattato è un grafico in *Figura 5.13* che mostra le differenze dei tempi rilevati:

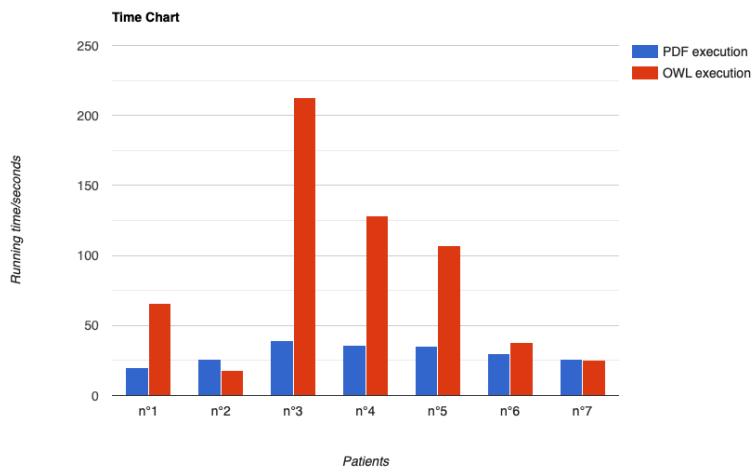


Figura 5.10: Differenze tempi esecuzione

Nel grafico soprastante sono rappresentati in **Rosso** i tempi relativi all’esecuzione del file *.owl* e in **Blu** i tempi del file *.pdf*. Si comprende subito che i tempi di esecuzione dei file *.owl* sono nettamente superiori, questo perchè il fulcro dell’architettura si basa si sull’estrazione dei dati, ma trova una complessità maggiore nel momento in cui si crea l’ontologia e la si popola secondo le informazioni estratte dalla cartella clinica digitale. Il caso specifico riguardante il popolamento dell’ontologia è stata trattata nei capitoli precedenti. Entrando invece nello specifico, si costruisce un grafico contenente le caratteristiche della cartella clinica relative ad ogni paziente preso in esame, la *Figura 5.14* mostra il risultato ottenuto:

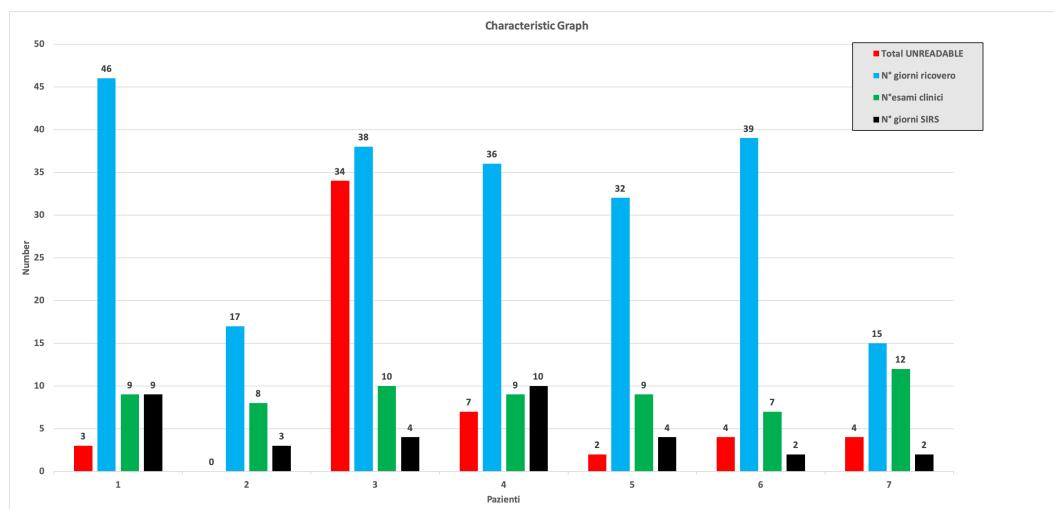


Figura 5.11: Grafico delle caratteristiche

Il grafico mette in evidenza le variazioni subite dal calcolo delle relative caratteristiche delle

cartelle cliniche, facendo notare che i giorni di ricovero non influiscono sul numero di esami praicati. Altro motivo di attenzione, lo si focalizza confrontando due metriche che a primo impatto lascerebbero pensare ad una forte dipendenza tra di loro, ma che gli studi e le rappresentazioni mostrate in *Figura 5.15* spiegano meglio:

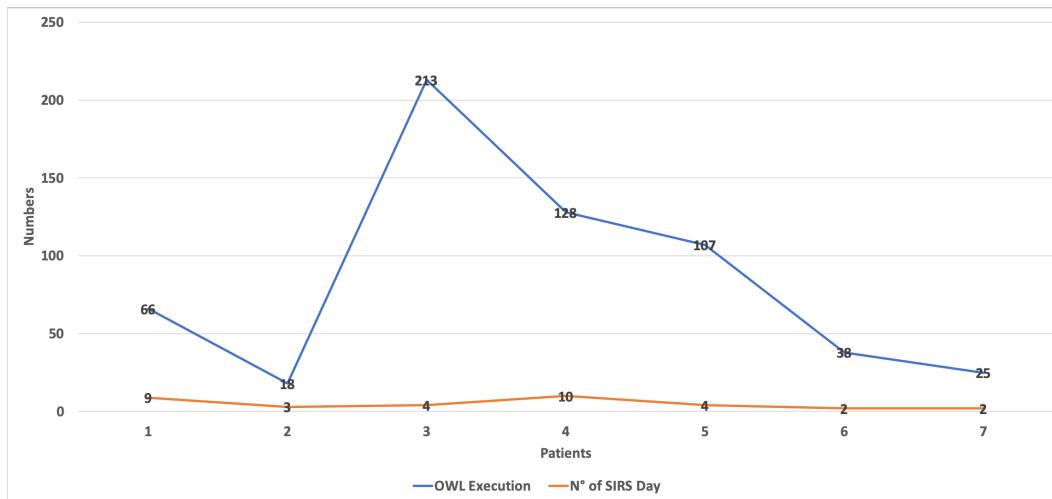


Figura 5.12: Relazione tra tempi OWL e numero giorni Sirs

Gli studi e le metriche calcolate, rappresentate poc’anzi mettono in relazione i tempi di esecuzione del file *.owl* con i giorni di *Sirs* calcolati. Alla luce dei precedenti risultati, ove mostrano le nette differenze di elaborazione tra i due file (*.pdf* e *.owl*), in un primo momento si supponeva che allora l’elevato tempo di esecuzione da parte del file ontologico, fosse dovuto ad un elevato numero di criteri gioranlieri che soddisfaccessero la *Sirs*. Motivo per cui si è ritenuto necessario studiare a fondo la casistica. I risultati ottenuti smentiscono che l’elaborazione sia strettamente collegata al numero di rilevazioni di *Sirs*, lo si nota specialmente nel caso del *Paziente 3* avente un tempo di esecuzione ontologico pari a circa 4 minuti (213 secondi) e un numero di rilevazioni *Sirs* pari a 4, mentre nel *Paziente 4* si nota come il tempo di esecuzione si strettamente inferiore poichè pari a circa 2 minuti e 13 secondi (128 secondi) ed un numero di rilevazioni *Sirs* nettamente superiori a quelle del *Paziente 3*. Si apprende dunque, che i tempi di esecuzione relativi al file ontologico, non hanno attinenza alcuna con le metriche calcolate, bensì esso varia a seconda della complessità del file *.xml* che si crea al momento dell’analisi. Il file ontologico è soggetto ad aggiornamento ma in questa sperimentazione non è stato possibile rendere tale. Così facendo il file ontologico sarà sempre relativo ad un solo paziente e il processo è strettamente collegato con la complessità della cartella clinica.

Il *Figura 5.16* seguente, invece mostra l’attinenza tra i tempi di esecuzione con il numero dei caratteri di una cartella clinica:

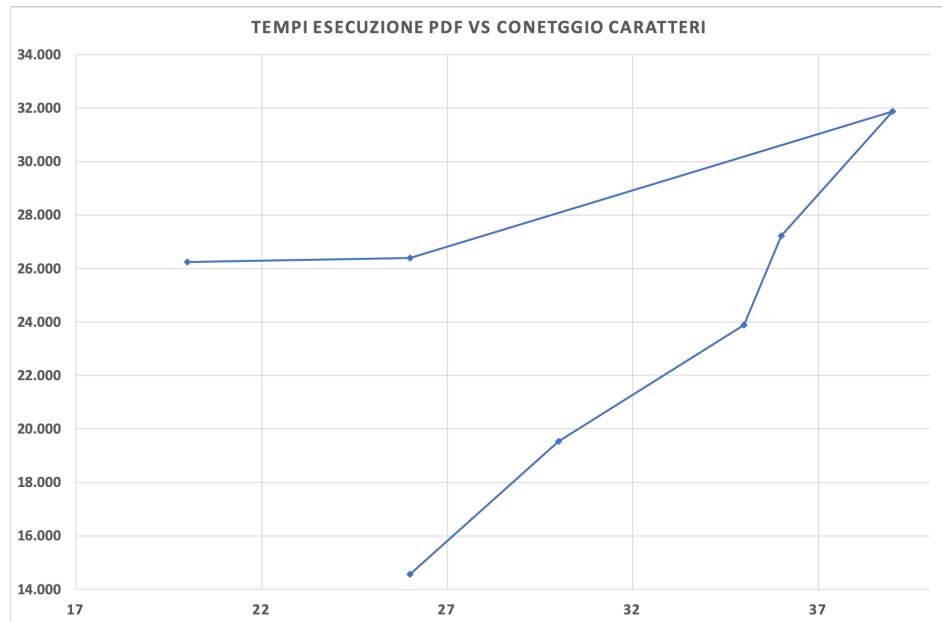
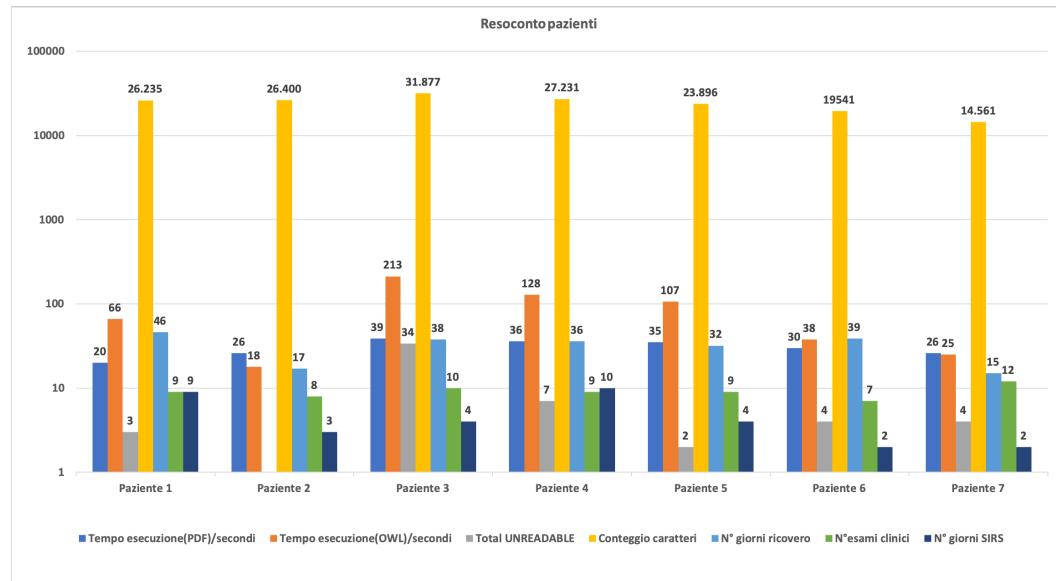


Figura 5.13: Tempi esecuzione VS conteggio caratteri

Sull’asse delle *x*, sono riportati i tempi di esecuzione del file *.pdf* o *.docx* (il conteggio dei caratteri è invariato a seconda del file che si sceglie di analizzare), mentre sull’asse delle *y* è riportato il range (da 14.000 a 34.000) di caratteri relativi ai *medical record* analizzati. Il grafico mostra chiaramente che man mano che il *medical record* diventa complesso, il tempo di lettura del file stesso, aumenta. Tuttavia si riscontra una contraddizione nel *Paziente 7* poichè il numero dei caratteri (si rimanda alla prima *tavella* mostrata) è di **14.561** che mostra un tempo di esecuzione pari a **26 secondi**: il motivo di questa contraddizione si basa sul fatto che durante la fase di ricovero, sono stati riportati quasi sempre solo i parametri vitali, esempio: HR (*Heart rate*) 110, ABP (*Arterial Blood Pressure*) 130/40, T (*Boody Temperature*) 36°C.

La *Figura 5.17* mostra un resoconto con tutte le metriche calcolate per ogni paziente.

**Figure 5.14:** Resoconto pazienti

5.4 La controprova

Una delle caratteristiche imprescindibili di ogni sistema ingegneristico, e non solo, dovrebbe essere l'*affidabilità*. In effetti nessuno acquisterebbe un'autovettura che in curva sia incline a capovolgersi o installerebbe un'ascensore che tende a bloccarsi, o tantomeno costruirebbe un ponte che oscilli spaventosamente, e così via. Anche nei sistemi software chiaramente si è interessati a questa caratteristica (purtroppo alcuni celebri errori in sistemi informatici, sono tragicamente passati alla storia) .Nell'ambito dei sistemi software, il termine di affidabilità implica due altri concetti:

- **Correttezza:** il sistema realizza le funzioni per il quale è stato progettato. Quindi a stimoli corretti di ingresso produce gli effetti di output previsti;
- **Robustezza:** il sistema è in grado di rilevare e gestire situazioni anomale (dati in ingresso non corretti, parti del sistema non funzionante, ecc.).

In parole povere, l'affidabilità è sintetizzabile con la minimazione del numero di errori (*bug*). Se poi nei sistemi informatici si vuole massimizzare la riusabilità del codice, si comprende come questa caratteristica assuma ancora più rilevanza.

Il problema da porsi è come mai sia così difficile realizzare sistemi affidabili, o meglio ancora, quale tecnica utilizzare per costruire sistemi affidabili. Come si può immaginare, la risposta non è univoca nè tantomeno semplice. Tuttavia, esistono delle *best practice* che permettono di

aumentare la qualità e l'affidabilità del software [17]. In questo contesto una delle *best practice* da poter svolgere equivale allo studio degli output forniti dal tool e di attuare una verifica per controprova. Visti i risultati sopra descritti, ogni cartella clinica sperimentata fornisce l'esito relativo a: **Paziente affetto da Sirs**, affiancata alla giustifica dell'esito stesso con: *codice fiscale paziente, data, orario e parametri per cui l'esito risulta positivo*. In vista di questi parametri è necessario effettuare:

- **Revisione** dei parametri che costituiscono Sirs attraverso l'applicativo *protégé*;
- **Revisione e modifica** cartella clinica del paziente;
- **Riesecuzione** processo di verifica Sirs, utilizzando la cartella clinica con le modifiche apportate;

La **revisione** dei parametri consiste nel verificare "quando" e "perchè" l'esito della sperimentazione equivale a dire che il paziente è affetto da **Sirs**. Successivamente, si passa alla **revisione e modifica** della cartella clinica digitale, secondo cui, i parametri precedentemente appresi dovranno essere modificati in modo tale da attribuire un esito negativo. L'ultima, ma non di sicuro per importanza, è la fase di **riesecuzione** della cartella clinica con le modifiche apportate. Ciò equivale a dire che il *medical record* sarà oggetto di una nuova sperimentazione, seguendo i passi indicati nei capitoli precedenti e di revisionare se effettivamente l'algoritmo effettua i giusti "confronti" con i criteri di Sirs.

Avendo eseguito con cura i precedenti passaggi su ogni cartella clinica affidata per la sperimentazione ne si ricava che: Il tool svolge correttamente la funzione di individuazione dei criteri di Sirs, poichè, avvalendosi delle necessarie modifiche ai criteri individuati nella prima sperimentazione e alla nuova esecuzione delle cartelle cliniche, i pazienti non mostrano forme di Sirs. La *Figura 5.18* e *Figura 5.19* hanno lo scopo di mostrare la fase di **revisione e modifica** della cartella clinica.

Daily Clinic Journal

SURNAME: **NAME:** **MEDICAL RECORD NUMBER:**

DATE	TIME	
	18.00	ABP 155/75 HR 108 bpm. T 37,4. Valid diuresis. Sedation and ventilation in IOT in increased BPAP. Sat Hb 99%.
16/11/2014	9.00	Patient sedated and ventilated in BIPAP. ABP 190/95 HR 108 bpm . T 38,1. With O2 100%.
	18.00	Patient not sedated in BIPAP mode. FiO2 0.5. Vital parameters: ABP 165/85 HR 95 bpm. T 38. Diuresis 1800ml from 8am. SpO2 98%. Hematochemical tests for tomorrow morning.
17/11/2014	9.00	Conscious patient, easily executes the commands. It has no engine deficits. It is intubated for X os and connected to VAM in CPAP with support. Pathological noises in the chest with (UNREADABLE) bibasale. Npt. Gas-open alair. Plasil 1 fl for 8 for three days. ABP 150/80 HR 100 bpm. T 37.9 diuresis in 24h 5300 ml. EGA in two hours.
	18.00	After analgesation and curation, VBS is carried out in VBS to practice tracheotomy according to blue rhino lashes. You place cannula 09.
	18.38	Patient in analgesation and siMV with Psupport. SatO2 100%. EOT - MUF. Sour but present. ABP 150/80 HR 87 bpm valid diuresis. He's preparing for exams.

Figure 5.15: Parametri con esito Sirs

In rosso sono evidenziati i due criteri che forniscono Sirs, che risultano essere: **Battico Cardiaco > 90 battiti al minuto e Temperatura Corporea > 38°C.**

Daily Clinic Journal		
SURNAME:	NAME:	MEDICAL RECORD NUMBER:
DATE	TIME	
	18.00	ABP 155/75 HR 108 bpm. T 37,4. Valid diuresis. Sedation and ventilation in IOT in increased BPAP. Sat Hb 99%.
16/11/2014	9.00	TEST WITHOUT SIRS
	18.00	Patient not sedated in BIPAP mode. FiO2 0.5. Vital parameters: ABP 165/85 HR 95 bpm. T 38. Diuresis 1800ml from 8am. SpO2 98%. Hematochemical tests for tomorrow morning.
17/11/2014	9.00	Conscious patient, easily executes the commands. It has no engine deficits. It is intubated for X os and connected to VAM in CPAP with support. Pathological noises in the chest with (UNREADABLE) bibasale. Npt. Gas-open alair. Plasil 1 fl for 8 for three days. ABP 150/80 HR 100 bpm. T 37.9 diuresis in 24h 5300 ml. EGA in two hours.
	18.00	After analgesation and curation, VBS is carried out in VBS to practice tracheotomy according to blue rhino lashes. You place cannula 09.
	18.38	Patient in analgesation and siMV with Psupport. SatO2 100%. EOT - MUF. Sour but present. ABP 150/80 HR 87 bpm valid diuresis. He's preparing for exams.

Figure 5.16: Modifica parametri

In rosso è stata evidenziata la gestione dei criteri di Sirs. Dopo la nuova esecuzione della cartella clinica, viene riportata la *Figura 5.20* che mostra l'effettiva veridicità del tool, non mostrando i criteri di Sirs.

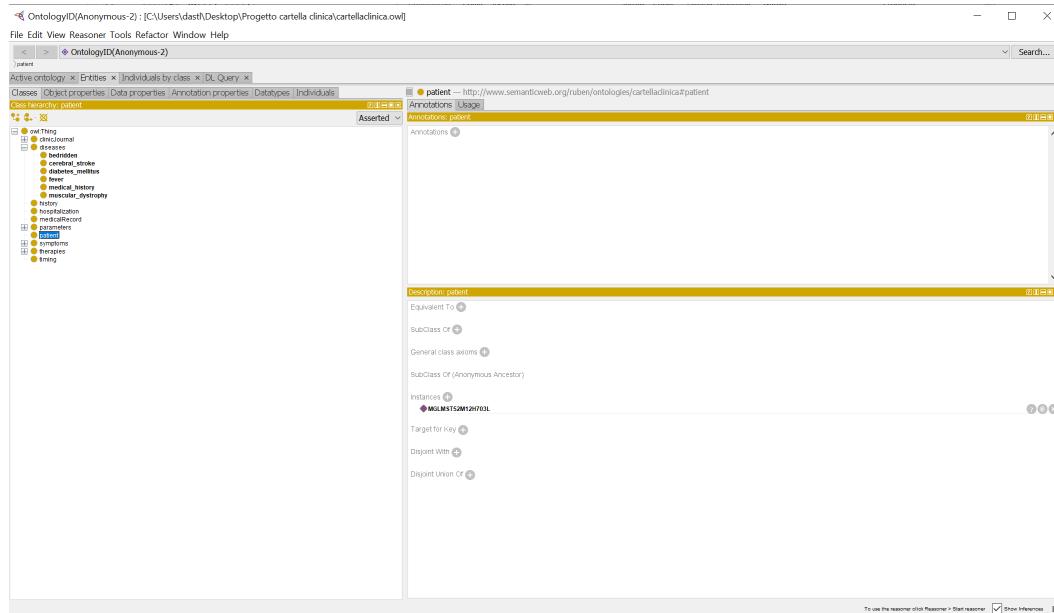


Figure 5.17: Nessun esito positivo per Sirs.

CHAPTER 6

Conclusioni e sviluppi futuri

In questa sezione si discute sui risultati ottenuti dando un giudizio complessivo di come il tool svolga il lavoro. Successivamente si esprimono le proposte a sviluppi futuri.

6.1 Conclusioni

Il tipo di approccio descritto in questo elaborato di tesi, si basa sullo studio e la sperimentazione di un tool che svolge il ruolo di supporter alle diagnosi di malattie critiche in ambito ospedaliero. Il tool oltre ad essere proposto in ambienti in cui si richiede una risposta pronta all'individuazione di malattie critiche, può essere proposto a qualsiasi reparto ospedaliero, in modo tale da poter essere costantemente utilizzato, prevenendo così la mancata diagnosi di **Sirs** e delle successive complicazioni che, spesso portano alla mortalità. La sperimentazione dunque riscontra un esito **positivo** per il seguente compito: **Individuazione ottimale dei criteri di Sirs per ogni cartella clinica digitale**, ed un esito **negativo** per: **Aggiornamento automatico dell'ontologia**, ovvero il mancato accrescere delle conoscenze a partire da una sola cartella clinica iniziale, poichè porta ad avere un'ontologia molto grande e ad aumentare i tempi di esecuzione che in ambito clinico risultano essere critici. In ambito clinico, gli sviluppi futuri sono innumerevoli, tra cui, quello avente priorità più alta è la possibilità di costruire moduli (interni o esterni) al tool già preso in considerazione che permettano di costruire la cartella clinica secondo i pattern e le validazioni necessarie al tool stesso. In modo tale da consentire ancora di più l'estendibilità del tool ai vari ambiti clinici poichè la cartella clinica

assumerebbe un concetto sempre più digitale, abbandonando definitivamente quello cartaceo che come abbiamo riscontrato, può causare problemi di comprensibilità.

Si potrebbe tentare un approccio diverso all'aggiornamento automatico dell'ontologia.

Si potrebbe aggiungere logica relativa all'individuazione di altre malattie e studiarne i relativi risultati. Infine, il tool potrebbe essere integrato in un sistema più ampio, che oltre all'analisi della cartella clinica, potrebbe integrare ulteriori informazioni, come quelle catturate dai sistemi di video-sorveglianza e dagli strumenti medici di monitoraggio dei pazienti. Un sistema di questo tipo permetterebbe il monitoraggio continuo dei pazienti rivocerati in pronto soccorso e/o in terapia intensiva, e potrebbe essere di forte ausilio per migliorare la diagnostica di malattie che, una volta sviluppate, quasi sempre portano alla degenerazione o morte in brevissimo tempo, come nel caso della Sepsi

Bibliography

- [1] AI4Business. Cos'è l'intelligenza artificiale, perchè tutti ne parlano e quali sono gli ambiti applicativi, 2019. <https://www.ai4business.it/intelligenza-artificiale/intelligenza-artificiale-cose/>.
- [2] M. Calci. La sepsi: nuova definizione ed evoluzione nei criteri di gestione. <http://www.itjem.org/articoli-scientifici/original-article/387-la-sepsi-nuova-definizione-ed-evoluzione-nei-criteri-di-gestione>.
- [3] W. W. Chapman e C. J. McDonald D. Demner-Fushman. What can natural language processing do for clinical decision support? *Journal of biomedical informatics*, 2009.
- [4] R. Leaman e G. Gonzalez. Banner: an executable survey of advances in biomedical named entity recognition. *in Biocomputing 2008*, pages 652–663, 2008.
- [5] S. Meystre e P. J. Haug. Natural language processing to extract medical problems medical problems from electronic clinical documents: performance evaluation. *Journal of biomedical informatics*, 39(6):589–599, 2006.
- [6] National Center for Biotechnology Information. Ncbi text mining tools. <https://www.ncbi.nlm.nih.gov/research/bionlp/Tools/>.
- [7] A. Pandey N. Arya G. Halford S. F. Jones R. Forshee M. Walderhaug e T. Botsis K. Kreimeyer, M. Foster. Natural language processing systems for capturing and standardizing unstructured clinical information: a systematic review. *Journal of biomedical informatics*, 2017.

- [8] C. Silva e R. Martinho L. Pereira, R. Rijo. Text mining applied to electronic medical records: A literature review. *International Journal of E-Health and Medical Communications (IJEHMC)*, 6(3):1–18, 2015.
- [9] MDCalc. Sirs, sepsis, and septic shock criteria. <https://www.mdcalc.com/sirs-sepsis-septic-shock-criteria>.
- [10] S. Abram C. G. Scott R. Chaudhry H. Liu I. J. Kullo e A. M. Arruda-Olson N. Afzal, S. Sohn. Mining peripheral arterial disease cases from narrative clinical notes using natural language processing. *Journal of vascular surgery*, 65(6):1753–1761, 2017.
- [11] S. Sohn H. Liu R. Chaudhry C. G. Scott I. J. Kullo e A. M. Arruda-Olson N. Afzal, V. P. Mallipeddi. Natural language processing of clinical notes for identification of critical limb ischemia. *International journal of medical informatics*, 111:83–89, 2018.
- [12] National Institutes of Health. Metamap - a tool for recognizing umls concepts in text. <https://metamap.nlm.nih.gov/>.
- [13] National Institutes of Health. Unified medical language system (umls). <https://www.nlm.nih.gov/research/umls/>.
- [14] R. Islamaj Dogan e Z. Lu R. Leaman. Dnorm: disease name normalization with pairwise learning to rank. *Bioinformatics*, 29(22):2909–2917, 2013.
- [15] R. Khare e Z. Lu R. Leaman. Challenges in clinical natural language processing for automated disorder normalization. *Journal of biomedical informatics*, 57:28–37, 2015.
- [16] A. Sparvoli. Sepsi: colpisce 700mila persone l'anno in europa, ma pochi sanno di che cosa si tratta, sep 2018. https://www.corriere.it/salute/malattie_infettive/cards/sepsi-colpisce-700mila-persone-l-anno-europa-ma-pochi-sanno-che-cosa-si-sistema-immunitario-fuori-controllo_principale.shtml.
- [17] L.Vetti Tagliati. *UML e ingegneria del software*. HOPS, 2003.
- [18] My Personal Trainer. Sepsi. <https://www.my-personaltrainer.it/salute/sepsi.html>.
- [19] Tutorialspoint. Apache poi word text extraction. https://www.tutorialspoint.com/apache_poi_word/apache_poi_word_text_extraction.htm.

- [20] Tutorialspoint. Pdfbox reading text. "https://www.tutorialspoint.com/pdfbox/pdfbox_reading_text.htm.
- [21] National Institutes of Health U.S. National Library of Medicine. Metamaplite. <https://metamap.nlm.nih.gov/MetaMapLite.shtml>.
- [22] P. Hanbury G. F. Cooper e B. G. Buchanan W. W. Chapman, W. Bridewell. A simple algorithm for identifying negated findings and diseases in discharge summaries. *Journal of biomedical informatics*, 34(5):301–310, 2001.
- [23] Wikipedia. Cartella clinica. https://it.wikipedia.org/wiki/Cartella_clinica.
- [24] Wikipedia. Ontologia. [https://it.wikipedia.org/wiki/Ontologia_\(informatica\)](https://it.wikipedia.org/wiki/Ontologia_(informatica)).
- [25] wikipedia. Protégé. [https://en.wikipedia.org/wiki/Protégé_\(software\)](https://en.wikipedia.org/wiki/Protégé_(software)).
- [26] Wikipedia. Sepsi. <https://it.wikipedia.org/wiki/Sepsi>.
- [27] M. Rastegar-Mojarad S. Moon F. Shen N. Afzal S. Liu Y. Zeng S. Mehrabi S. Sohn e H. Liu Y. Wang, L. Wang. Clinical information extraction applications: a literature review. *Journal of biomedical informatics*, 77:34–49, 2018.
- [28] X. Li-T. Naumann e Y. Luo Z. Zeng, Y. Deng. Natural language processing for ehr-based computational phenotyping. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(1):139–153, 2019.

Ringraziamenti

Innanzitutto vorrei ringraziare il Prof. Vincenzo Deufemia e la dott.ssa Loredana Caruccio per avermi dedicato tempo e pazienza. Persone disponibili e socievoli, che mi hanno guidato durante tutto lo sviluppo del progetto.

Ed eccomi qui.. Dopo le scuole superiori non avrei mai pensato di proseguire il mio percorso di studi all'Università. Università: una parola che per me rappresentava la maestosità ed io a confronto ero solo una piccola pedina, che non sarebbe mai stata in grado di affrontarla...e invece, sono qui con le persone che mi vogliono bene a festeggiare questo piccolo traguardo. Piccolo sì, perchè finalmente ho capito di avere la "stoffa" per perseguire i miei sogni e di puntare sempre più in alto.

Voglio ringraziare i miei genitori, che hanno investito su di me, nonostante in passato gli abbia recato molti dispiaceri. Spero possiate essere fieri di me, per questo piccolo traguardo e, spero di aver rimediato a tutte quelle volte che vi ho fatto disperare. Grazie per aver avuto fiducia e pazienza. Pazienza sì, perchè con me ce ne vuole tanta e ne sono consapevole. Se mi dovessero chiedere :"come vorresti essere da grande?" risponderei : "Come i miei genitori". A mio padre: mentore e modello che ho sempre perseguito. A mia madre: fonte di saggezza e speranza ma soprattutto sempre pronta a difendermi. A mia sorella: spalla su cui piangere e gioire. A voi devo tutto ciò che sono...ancora GRAZIE.

Vorrei ringraziare il mio gruppo di studio, il team **DELTA CAPPELLO** che nel corso di questi tre anni è diventato sempre più grande. Un gruppo di ragazzi meravigliosi, sempre pronti ad aiutare senza mai chiedere nulla in cambio. Se questa esperienza univeristarria la considero meravigliosa, è anche per merito loro. Spero che nella vita, tutti possano avere

un gruppo di studi del genere e, come tale desidero ringraziarvi singolarmente: Gilberto, Raffaele, Nicola, Emilio, Andrea Califano, Cristiana, Stefano, Alfonso, Gaetano, Luca, Donato, Nello, Gianluca, Simone, Francesco, Giovanni, Luigi, Carmine, Vincenzo, Mimmo, Adalgiso e Gerardo. Siete fantastici, grazie di tutto.

In particolare, vorrei ringraziare il mio migliore amico Simone, che ha sempre creduto in me, anche quando io stesso avevo perso le speranze. Sei l'amico che tutti dovrebbero avere, il vero amico, quello che c'è sempre stato senza mai battere ciglio, a te devo tutto. Grazie Simo. Ringrazio Arianna e Daniele, fedeli compagni che mi accettano e che mi vogliono bene nonostante io li abbia lasciati soli, per via dello studio. Sappiate che non mi sono mai dimenticato di voi.

Ringrazio questo splendido percorso univeristario perchè tra tante insidie e tanti ostacoli, mi ha fatto incontrare una persona speciale che, da un anno a questa parte mi segue incodizionatamente, senza mai lasciarmi solo, MAI. Con lei ho condiviso tutto, le ansie, i pianti, le gioie e le serate passate a ballare. Grazie di tutto Cristiana, ti amo!

Questa è una giornata meravigliosa e lo è per tanti motivi, perchè ci sono i miei amici, c'è la mia famiglia e infine c'è il cielo. Quando c'è il cielo è un po diverso perchè tutte le persone che ho perso nella mia vita, ti possono guardare finalmente. Caro Nonno Manfredi e cara Nonna Teresa, come vedete, ho mantenuto la mia promessa, spero possiate essere fieri di me, come lo sono io di voi. Vi amo!

Ringrazio me stesso, per averci creduto sempre, nonostante le difficoltà e ringrazio la mia forza di volontà. *"Serve la tentazione di mollare tutto, per decidere.. di riprovare ancora"*. Non chi comincia, ma quel che persevera.