UNIVERSITÀ DI PISA

Corso di Laurea in Informatica Umanistica

RELAZIONE

Information Extraction applicata a testi storici

Candidato: *Federica Semplici*

Relatore: *Alessandro Lenci*

Correlatore: *Felice Dell'Orletta*

Anno Accademico 2014-2015

Indice Generale

1. Introduzione

Di fronte ai nuovi scenari che si presentano nell'ambito delle Digital Humanities è facile lasciarsi andare a fantasie di applicazioni grandiose e ritrovarsi a parlare con *HAL 9000*, il supercomputer del film *2001: Odissea nello spazio* dotato di Intelligenza Artificiale e in grado di interagire completamente con l'uomo, parlare, ragionare e compiere scelte appropriate.

Nonostante questi scenari fantastici, anche se ancora molto lontani dalla realtà e dallo stato dell'arte, non dobbiamo dimenticare ciò di cui facciamo parte e da cui proveniamo. Questo progetto fa parte di un ambizioso tentativo di porre attenzione alla memoria storica sfruttando tecniche di trattamento automatico della lingua applicate a documenti che raccontano l'Italia nel corso dei due conflitti mondiali: Memorie di Guerra[1] è un portale dedicato alla Prima e alla Seconda Guerra Mondiale, in cui è possibile trovare i bollettini di guerra, i diari dei soldati e di chi ha vissuto in qualche modo i due conflitti, analizzati e osservabili secondo la linea del tempo, dello spazio, dei termini utilizzati e dei macro-eventi o grandi battaglie individuate dagli storici.

Il progetto nasce in occasione delle celebrazioni del centenario dall'inizio della Prima Guerra Mondiale e dei 70 anni dalla conclusione della Seconda Guerra Mondiale, grazie all'Università di Pisa e l'Istituto di Linguistica Computazionale A.Zampolli, in collaborazione con l'Università di Siena.

Con l'unione tra informatica, linguistica e storia questo lavoro si colloca perfettamente nell'ambito delle Digital Humanities. Gli obiettivi sono molteplici e spaziano dal preservare i testi, alla loro diffusione, valorizzazione e analisi sotto nuovi punti di vista, sfruttando tecnologie informatiche e, in particolare, strumenti di linguistica computazionale e un'interfaccia grafica per l'esplorazione dei dati da parte dell'utente.

Il progetto a cui ho lavorato consiste in primo luogo nell'individuazione automatica degli eventi bellici presenti nei bollettini di guerra italiani della Prima e delle Seconda

[1] Il sito del progetto *Memorie di Guerra* è consultabile all'indirizzo http://www.memoriediguerra.it/wwm/

Guerra Mondiale. Gli eventi vengono descritti da termini o sequenze più complesse, inserite in un'ontologia creata manualmente. In una seconda fase, vengono estratte dal testo le Entità Nominate di tipo Luogo.

L'obiettivo finale è individuare le relazioni "è avvenuto a" in modo automatico tra le entità Eventi e Luoghi. Per fare questo sono stati tentati due diversi procedimenti: stabilire le relazioni in base a regole di adiacenza tra entità e in base a regole di dipendenza sintattica. E' stato utilizzato un approccio basato su regole in quanto la mancanza di un corpus annotato non consentiva l'uso di algoritmi di apprendimento automatico.

Per quanto riguarda la struttura di questo elaborato, nel capitolo 2 "Information Extraction" vengono brevemente descritti i diversi metodi di approccio possibili per svolgere compiti di estrazione di entità e definizione di relazioni. Ci concentreremo in particolare sulle basi teoriche utilizzate nello svolgimento del progetto e anche sugli aspetti caratteristici di procedimenti non utilizzati per chiarire perché siano stati scartati.

Nel capitolo 3 "I Testi" vengono descritti i bollettini di guerra, dalle caratteristiche linguistiche alle prime fasi di elaborazione per arrivare alla forma del testo usato come input del progetto.

Il capitolo 4 "Fasi di Analisi" descrive come sia stato portato avanti il progetto da un punto di vista pratico: il popolamento dell'ontologia e le sue caratteristiche, gli script per l'estrazione delle entità e delle relazioni, l'output.

Nel capitolo 5 "Valutazione delle fasi di Analisi" vengono riportati i dati relativi alle performance ottenute, divisi per fasi di analisi e per corpus (Prima e Seconda Guerra Mondiale distintamente). Infine le conclusioni.

2. Information Extraction

Il progetto trattato in questa tesi di laurea pone le sue basi teoriche sul concetto di *Information Extraction* (IE). In questo capitolo verranno descritte alcune delle tecniche più utilizzate per estrarre entità nominate e relazioni, ciò che è necessario per la realizzazione pratica del progetto, e le relative misure di valutazione.

Per IE s'intende, nell'ambito del Tal (*Trattamento Automatico del Linguaggio*) il compito di estrazione automatica e identificazione di alcuni tipi di entità, relazioni o eventi in un testo. I task inclusi sono molti, tra i quali spiccano l'identificazione e la classificazione di Entità Nominate (*Named Entity Recognition and Classification*, NERC o NER) e *Relation Detection and Classification* ovvero l'estrazione e la classificazione di relazioni ed eventi.

2.1 Basi storiche e importanza dei task di IE

I primi studi risalgono agli anni '70 e proseguono fino ad oggi, intensificati e resi sempre più necessari dalla mole di dati digitalizzati presenti sul web. Gli algoritmi di IE sono infatti la risposta alla crescita esponenziale di testi disponibili: un ottimo sistema di estrazione di informazione ci permetterebbe di sapere in anticipo di cosa tratta un documento senza doverlo leggere. Sapere a priori se ci può interessare o meno può risparmiarci grandi quantità di tempo. Inoltre estrarre la conoscenza contenuta in un testo in modo automatico, può essere utilizzato in ambiti applicativi per compiti di ranking o indicizzazione, di gestione e organizzazione della conoscenza, linking a risorse esterne, ecc.

Il valore di un simile compito è reso ancora più evidente dall'interesse dimostrato dal governo degli Stati Uniti d'America che incentivò la discussione sul tema a partire dalle conferenze MUC (*Message Understanding Conference*)[2] degli anni '90[3].

[2] Conferenze a carattere competitivo in cui vengono presentati i risultati ottenuti in ambito di IE
[3] Piskorski & Yangarber (2013)

Gli sforzi più recenti in campo di IE si rivolgono alla *sentiment analysis*, basata per lo più su contenuti estratti da social media come blog, forum, tweet. L'obiettivo è definire in modo automatico se l'opinione espressa dall'utente è positiva o negativa, un compito che si sta rivelando molto utile per fini applicativi soprattutto nell'ambito della politica e del marketing.

2.2 Named Entity Recognition

La prima fase prevista in un compito di IE è solitamente il riconoscimento e la classificazione dei nomi propri. Nelle applicazioni più comuni si tratta di nomi di persona, di luogo e di organizzazioni ma, a seconda del tipo di testo, possiamo avere la necessità di estrarre altri tipi di elementi come nomi di geni, proteine, elementi chimici se stiamo analizzando un testo di chimica, medicina o biologia, nomi e marche di automobili, titoli di opere d'arte o di film. Dovremo quindi trovare i token o le porzioni di testo che costituiscono nomi propri e successivamente classificarli per tipo semantico.

L'identificazione di questi elementi è un compito fondamentale per l'analisi linguistica ed è basata sul modo in cui viene rappresentato l'elemento che stiamo ricercando in una data lingua o tipo di testo; in particolare possono essere determinanti le caratteristiche linguistiche del testo e delle entità che vogliamo estrarre e l'appartenenza a uno specifico dominio di interesse. [4]

Possiamo utilizzare metodi di apprendimento automatico o regole e espressioni regolari per definire pattern che identifichino un determinato gruppo di entità in quanto possiedono particolari caratteristiche o *features*.

Le features possono essere:

- locali: forma e lunghezza del token, presenza della punteggiatura all'interno del token (acronimi *C.N.R.*), alternanze atipiche (*eBay*), la posizione all'interno della frase, la presenza della parola all'interno di liste di token con altissima

[4] Mitkow(2004)

probabilità di essere NE dei diversi tipi (liste di nomi di luoghi con le relative indicazioni geopolitiche e dettagli geografici sono chiamati *gazetters*);

- contestuali: valutazione del token precedente e successivo (*Sig. Rossi*, *Ferrero S.p.A.*), caratteristiche tipografiche della parola successiva (iniziale maiuscola);
- globali: il tipo del documento.

2.2.1 Entità Nominate e Ambiguità

Nella fase di classificazione di NE possiamo riscontrare ambiguità di due tipi: una istanza può essere riconducibile a due diverse entità dello stesso tipo o a entità di tipo diverso. Il caso di "*JFK*" è significativo e presenta entrambi i tipi di ambiguità: l'istanza può riferirsi al presidente americano John Fitzgerald Kennedy ma anche al figlio, in entrambi i casi entità "Persona", ma "*JFK*" è anche il nome dell' aeroporto di NewYork e di alcune scuole, ponti e strade negli Stati Uniti e non solo, tutte entità di tipo diverso. La disambiguazione può avvenire in base al contesto, sfruttando algoritmi che utilizzano la distribuzione statistica e le co-occorrenze dei termini.[5]

2.2.2 Entità Nominate e XML

Il risultato dell'identificazione della NE può essere rappresentato con un linguaggio di markup, spesso XML, con un tag in apertura del tipo <NAME TYPE = " xxx"> e il tag </NAME> in chiusura.

In una frase come:

Galileo Galilei è nato a Pisa.

Avremo quindi:

<NAME TYPE = "PERSON">*Galileo Galilei*</NAME> *è nato a* <NAME TYPE="LOCATION"> *Pisa*</NAME>

[5] Jurafsky & Martin (2006)

2.2.3 Entità Nominate e IOB format

Possiamo considerare il compito di classificazione di NE come un problema di etichettatura. Si dovrà quindi trovare il confine dell'entità e apporre la marca corretta. La presenza di una NE viene marcata con tag nel formato IOB in cui il tag B (begin) indica l'inizio dell'entità, I (internal) indica un token all'interno dell'entità e O (outside) definisce un token esterno, ovvero non è entità.

Ad ogni token viene assegnato un tag IOB e, se entità, anche il tipo di NE a cui appartiene. Analizzando la frase dell'esempio precedente avremo:

Galileo	B-Pers
Gailei	I-Pers
è	O
nato	O
a	O
Pisa	B-Loc

2.2.4 Metodi per l'estrazione e il riconoscimento di NE

Possiamo identificare tre tipi diversi di approcci per costruire un riconoscitore e classificatore di Entità Nominate. Questi utilizzano:

- algoritmi di apprendimento automatico
- espressioni regolari, regole e pattern
- liste di Entità Nominate (gazetteers)

I metodi che sfruttano algoritmi di apprendimento automatico prevedono due fasi di elaborazione. L'obiettivo della prima fase è la creazione di un modello statistico della distribuzione delle entità nel testo. Il modello viene estratto, tramite algoritmi di apprendimento supervisionato, da un training corpus annotato manualmente con le corrette classi IOB di output. Il corpus deve essere di grandi dimensioni e rappresentativo del testo su cui sarà applicato il modello.

La seconda fase si occupa dell'analisi vera e propria, applicando gli algoritmi e il modello al testo che vogliamo analizzare. Le features vengono estratte ed elaborate ("pesate") statisticamente per poi definire la classe di output più probabile per ogni token.

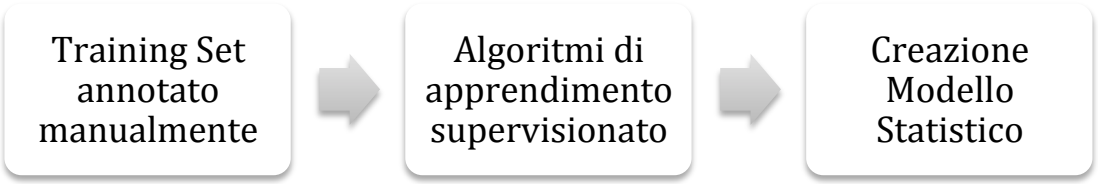


Figura 1: Fase di addestramento

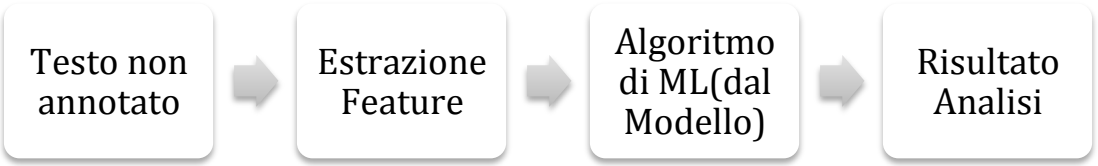


Figura 2: Fase di analisi

Gli algoritmi di machine learning (ML) richiedono una grande mole di dati annotati manualmente. Pertanto molto spesso non è possibile utilizzarli perché il corpus non esiste, ed è complesso, lungo e costoso crearlo.

Un altro metodo automatico, con un training corpus di dimensioni limitate annotato manualmente e un piccolo set di NE viene chiamato *bootstrapping*. Il procedimento consiste nel proiettare il set sul testo non annotato ed estrarre successivamente le features che definiscono quando un token è una NE in modo automatico. Il processo è iterativo: una volta estratte le features, queste vengono riutilizzate sul testo per individuare le NE non presenti nel set iniziale. Per svolgere correttamente il compito il testo non annotato deve essere di grandi dimensioni in modo che possano essere osservate e "pesate" più contesti possibili.[6]

Per ovviare al problema della mancanza di training corpus si possono sfruttare gli altri

[6] Jurafsky & Martin (2006)

due tipi di approccio: utilizzare regole o liste.

L'utilizzo di regole ed espressioni regolari consente di ottenere buone performance ma necessita di un esperto di IE o linguista che le definisca manualmente, e, se il compito o il testo cambiano drasticamente, andranno modificate le regole, con problemi di adattamento più lunghi di quelli con un sistema di machine learning.[7]

La gestione delle NE solo con liste è ovviamente molto limitante: andrebbero aggiornate continuamente, per molti tipi di entità o domini non esistono e rimarrebbe comunque il problema delle ambiguità da risolvere.

Spesso gli strumenti di NER cercano di combinare liste, regole e algoritmi di machine learning supervisionato[8] per ottenere migliori risultati. Resta comunque fondamentale che le varie componenti siano ben adattate al tipo di testo che si andrà ad analizzare e alle sue peculiarità.

2.3 Event Extraction

Per Event Extraction si intende il compito di estrazione di eventi dal testo. Estraendo le Entità Nominate possiamo identificare alcune componenti fondamentali dal punto di vista informativo all'interno di un documento. Tuttavia per estrarre informazioni più precise e ottenere un quadro completo della situazione descritta diventa necessario che il computer sia in grado di identificare l'evento a cui prendono parte le NE e creare una sorta di linea temporale in cui collocare i vari eventi descritti.

L'individuazione delle espressioni temporali viene denominata *Temporal Expression Recognition* e consiste nell'identificazione di espressioni temporali assolute come date e orari (*il 3 Aprile*, *alle 17.00*), espressioni che indicano la durata di un evento (*per tre ore, due settimane*) ed espressioni temporali relative, che descrivono un avvenimento rispetto ad un altro o al momento dell'enunciazione (*il giorno prima, ieri*). La fase successiva *Temporal Normalization*, consiste nel mappare le espressioni temporali

[7] Mitkow (2004)
[8] Jackson and Moulinier (2002)

estratte posizionandole su una linea del tempo e definendo i rapporti di anteriorità/posteriorità tra gli eventi a cui sono collegate.

L'estrazione di istanze che descrivono una determinata classe di eventi può essere eseguita con procedimenti basati su regole o algoritmi di apprendimento automatico. Entrambi i metodi si basano sulle informazioni estratte nei precedenti livelli di analisi, in particolare la marca indicata dal Pos Tagger, il tempo verbale e la presenza di termini particolari.

Spesso la descrizione di eventi è demandata al verbo (*bombardare*) ma possiamo trovare anche sostantivi (*bombardamento*) o sequenze composte da più token (*sganciare bombe, lancio di bombe*). Inoltre non tutti i verbi indicano la presenza di un evento, altri come *essere, stare* e *avere* possono avere funzione di ausiliare, altri ancora cambiano di significato a seconda del contesto a cui si accompagnano (*lasciar cadere* - *lasciar cadere bombe*) rendendo la classificazione più complessa.

Nonostante il task di Event Extraction sia argomento di numerose ricerche, a causa della sua complessità non sono ancora stati raggiunti risultati del tutto soddisfacenti.[9]

2.4 Relation Detection and Classification

Il compito di Relation Detection and Classification consiste nel trovare e classificare le relazioni semantiche che sussistono tra entità in un testo, molto spesso entità nominate ma non obbligatoriamente. In questo caso è molto importante il task di estrazione e riconoscimento delle stesse in quanto gli errori si ripercuoterebbero a cascata.

Nella maggior parte dei casi vengono individuate relazioni di tipo binario, queste possono essere di tipo tassonomico ma anche altri tipi definite dal linguista e legate al testo o al dominio specifico. Alcune relazioni semantiche che è possibile ricercare sono la meronimia[10], relazioni geospaziali, attributi di un elemento (colore).

Basandosi su relazioni binarie il task si avvicina al compito di popolare un database

[9] Jurafsky and Martin (2006)
[10] *X* è meronimo di *Y* se *X* è parte di/ membro di *Y*

relazionale: verranno quindi definiti i due argomenti, il ruolo di ognuno nella relazione e l'etichetta che indica il tipo di relazione.

2.4.1 Metodi di RD

In sistemi di apprendimento automatico il procedimento è simile a quello descritto per i sistemi di NER: il testo viene prima annotato manualmente con le relazioni semantiche che vogliamo identificare tra i diversi elementi e poi utilizzato come training corpus per allenare il sistema, sfruttando le feature estratte. Le feature considerate sono in primo luogo quelle relative alle NE, in particolare il tipo delle entità candidate come argomenti della relazione e l'osservazione del testo tra i due candidati (distanza e numero di entità tra argomenti, distanza sintattica nell'albero, costituenti, presenza di strutture predittive della presenza di una relazione). Per l'estrazione delle features relative al livello sintattico spesso si fa affidamento su strumenti di "*partial parsing*" che individuano solo alcuni tipi di costituenti di base, risparmiando tempo rispetto a strumenti di "*full parsing*", ma restituendo indicazioni adeguate a svolgere il compito.

Realizzando il sistema tramite espressioni regolari vengono definiti segmenti di testo che possano contenere le entità e le relazioni che vogliamo estrarre. Tuttavia creare tutti i pattern possibili è molto complesso a causa del gran numero di costruzioni diverse con cui possiamo esprimere uno stesso concetto. L'approccio con espressioni regolari consente di creare una serie di pattern molto specifici, o, al contrario, generalizzare i pattern, nonostante si rischi di creare falsi positivi.[11]

Un esempio:

"La FIAT ha uno stabilimento a Mirafiori… "

se volessimo estrarre la relazione "ha uno stabilimento a" tra le due entità "organizzazione" e "luogo" dovremmo definire una serie di pattern che la identifichino:

/[Org] ha uno stabilimento a [Luogo]/

[11] Mitkow (2004)

ma anche

/Lo stabilimento di [Luogo] di [Org]/	Lo stabilimento di Mirafiori della FIAT
/Lo stabilimento di [Org] di [Luogo]/	Lo stabilimento della FIAT di Mirafiori
/Lo stabilimento di [Org] a [Luogo]/	Lo stabilimento della FIAT a Mirafiori

Come per processi di NER è possibile utilizzare il metodo di bootstrapping con un piccolo set di "*seed pattern*" o "*seed tuple*" da cui estrarre le altre. Nell'esempio precedente utilizzando la tupla "FIAT – Mirafiori", si possono verificare tutte le occorrenze della coppia ed estrarre i nuovi possibili pattern. Alla fine del processo iterativo viene verificata la correttezza e affidabilità dei pattern individuati.

2.5 Valutazione di Sistemi di NER

Le misure utilizzate per valutare sistemi di Estrazione di Entità Nominate sono *Precision*, *Recall*, e *F-measure*. E' necessario disporre di un *Gold Standard,* un test set annotato manualmente con le corrette classi di output con cui confrontare le risposte ottenute dal sistema che stiamo valutando.

La Precision misura l'esattezza del sistema in percentuale. In formule:

$$P = \frac{Output\ Corretti}{(Output\ Corretti + Output\ Errati)} \times 100$$

La Recall misura la completezza o copertura del sistema in percentuale. In formule:

$$R = \frac{Output\ Corretti}{(Output\ Corretti + Output\ Mancati)} \times 100$$

Per output corretti si intende il numero di output individuati e etichettati correttamente

dal sistema (true positive), Output Errati indica il numero di output diversi da quelli presenti nel gold standard (false positive), Output mancati è il numero di situazioni che il sistema avrebbe dovuto individuare ma non ha riconosciuto (false negative).

Precision e Recall sono due misure in competizione tra di loro: molto spesso cercando di aumentare la copertura dell'algoritmo si aumentano esponenzialmente gli errori, diminuendone la precisione, e, al contrario cercando di diminuire il numero di output errati il rischio è quello di aumentare gli output mancati. Per avere una visione completa e indicazioni significative sulle performance del sistema di analisi, i due valori devono sempre essere presentati insieme.

In alternativa è possibile calcolare la F-measure[12], un metodo che consente di combinare insieme Precision e Recall in un'unica misurazione. In formule:

$$F = \frac{2\,PR}{(P+R)} \times 100$$

Se precisione e recall sono bilanciate il calcolo prevede la media armonica tra P e R.

Per quanto concerne lo stato dell'arte, un buon sistema di NE Recognition and Classification per la lingua italiana riesce ad ottenere risultati che si attestano attorno all'85% per Entità Geopolitiche, 88% per Persone, 51% per Luoghi, 70% per le Organizzazioni e 82% in totale, misurate con sistema F-misure[13].

2.6 Valutazione di sistemi di RD

La valutazione di sistemi di Relation Detection è simile ma ovviamente più complessa, a causa del numero di parti in gioco, rispetto alla valutazione di sistemi di NER.

Possiamo individuare due diversi tipi di approccio che sfruttano Gold Standard,

[12] van Rijsbergen (1975)
[13] Dati rilevati durante la campagna di Evalita 2009. http://www.evalita.it/2009/results

Precision, Recall e F-Measure.

Nel primo caso viene confrontato l'output prodotto dal sistema con il Gold Standard e ne vengono calcolate Precision, Recall e F-Measure. La procedura è duplice se vogliamo considerare sia l'individuazione della relazione tra le due entità, che la corretta etichettatura assegnata alla relazione stessa. Con il doppio valore (con e senza etichetta) per ogni misurazione possiamo ottenere informazioni più significative sulle capacità e limiti del nostro sistema.

Il secondo tipo di approccio prevede una verifica sul database finale, invece che sulle singole istanze estratte. Ciò significa che le istanze delle relazioni vengono inserite in un dataset e non vengono duplicate se già presenti. In questo modo si focalizza l'attenzione su ciò che è stato estratto globalmente dal testo , non importa in quale punto preciso è stata estratta la relazione né quante volte. Dopo aver popolato il database vengono calcolate precision, recall e F-measure confrontando i dati con il database estratto, con lo stesso procedimento, dal Gold Standard. Questo approccio prevede, generalmente, percentuali di valutazione migliori.

3. I Testi

I testi analizzati in questo progetto sono i bollettini di guerra dei due conflitti mondiali, emessi giornalmente dal Comando Supremo Italiano.

Le edizioni utilizzate sono:

- *I Bollettini della Guerra 1915-1918*, Alpes, Milano, 1923
- *Bollettini di guerra del Comando Supremo: 1940-1943*, Stato maggiore dell'Esercito, Ufficio Storico, Roma, 1970

I bollettini della Prima Guerra Mondiale sono 1342, in formato cartaceo, datati sequenzialmente dal 24 maggio del 1915 fino all'11 novembre 1918, data della fine della guerra in Europa[14]. Sono incluse la dichiarazione di guerra all'Impero Austro-Ungarico in testa, seguita, in tempi successivi, dalle dichiarazioni di guerra all'Impero Tedesco, Impero Ottomano e Regno di Bulgaria, e le dichiarazioni di fine delle ostilità con il proclama di Re Vittorio Emanuele III alle truppe e gli armistizi con le diverse potenze implicate nel conflitto. I bollettini della Seconda Guerra Mondiale, già in formato digitale, sono 1201, datati a partire dal 10 giugno 1940 fino all'8 settembre del 1943. .In ogni bollettino vengono descritti gli esiti delle attività belliche portate avanti nella giornata. Venivano pubblicati quotidianamente sui giornali e, durante il secondo conflitto, divulgati anche via radio.

Questi testi rappresentano la voce ufficiale del Governo e dell'Esercito e per questo motivo sono inevitabilmente parziali, non oggettivi, propagandistici. Per questo motivo spesso venivano taciuti o minimizzati il numero dei caduti o i danni subiti dall'esercito o dalle città italiane mentre, al contrario, ne venivano enfatizzate le vittorie e il numero dei prigionieri catturati. Altri eventi non venivano menzionati per motivi di segretezza bellica.

[14] in Italia si era già conclusa il 4 novembre 1918.

3.1 Digitalizzazione dei testi

I bollettini della Prima Guerra Mondiale non erano mai stati digitalizzati. La conversione è stata effettuata tramite scansione del documento, uso di un software *Optical Character Recognition* (OCR) e successiva revisione manuale per raggiungere un livello di correttezza ottimale ed evitare che gli errori si propagassero a cascata sui livelli di analisi successivi.

In un secondo momento, i testi sono stati convertiti in formato XML (*Extendible Markup Language*) mantenendo le indicazioni originali relativamente a paragrafi e spaziature e sono stati inseriti alcuni metadati.

Il testo del bollettino XML si trova racchiuso tra due marche "doc"; in apertura vengono indicati come attributi dell'elemento i seguenti metadati:

- *url*: L'url del documento html in cui si trova il bollettino
- *index*: un numero progressivo di indice
- *day*: un numero progressivo indicante da quanti giorni è iniziata la guerra
- *date*: la data completa
- *firma*: il nome del firmatario del bollettino, quasi sempre il Capo di Stato Maggiore in carica in quel momento.

il bollettino viene chiuso con la marca </doc>

Un esempio di marca di apertura:

<doc url="http://www.ilc.cnr.it/w2m/doc494.html" index="495" day="472" date="5 settembre 1916" firma="CADORNA">

3.2 Aspetti linguistici

Da un punto di vista linguistico emergono due aspetti fondamentali: la provenienza della maggior parte dei termini da un unico dominio e la distanza dall'italiano standard contemporaneo.

Analizzando il primo aspetto è facile accorgersi della ricorrente presenza di termini come “assalto”, “attacco”, “distruzione”, “esplosione”, provenienti dal dominio bellico, indipendentemente dalla categoria morfosintattica (per lo più nomi e verbi). Ogni bollettino contiene la descrizione di bombardamenti, conquiste, avanzate, ritirate, incursioni, ecc.. Sono anche presenti nomi di generali, aviatori, battaglioni, reggimenti, aerei, navi e nomi di luogo (v. cap. 3.4).

Il secondo aspetto rilevante, ovvero la distanza dall’italiano standard contemporaneo risulta ovvio dato il contesto storico. Nel 1915 l’Italia è unificata ma ancora in fase di formazione come nazione e con essa anche l’italiano come lingua nazionale. Non esiste ancora una vera e propria lingua diffusa, parlata e compresa da tutti e in alcune parti d’Italia l’utilizzo del dialetto è ancora la norma.

Nell’arco dei 30 anni di guerra possiamo assistere ad una evoluzione non solo dei metodi di combattimento e delle strategie di propaganda ma anche della stessa lingua con cui sono scritti i testi.

Possiamo trovare differenze sostanziali con lo standard, ovviamente molto più accentuate nei testi della Grande Guerra, sotto diversi livelli di analisi linguistica.

- termini non più utilizzati o desueti (verbo *fugare*)
- termini la cui resa grafica oggi non è accettata come corretta (*diecina* per *decina*, *schiatori* o *skiatori* per *sciatori*)
- costruzioni sintattiche desuete, latineggianti o classicheggianti

3.3 Prime fasi di elaborazione

I bollettini sono stati digitalizzati e successivamente annotati automaticamente a vari livelli di analisi linguistica:

- sentence splitting: divisione in frasi

- tokenizzazione: riconoscimento e divisione in tokens

- lemmatizzazione: ogni token viene ricondotto al proprio lemma

- annotazione morfosintattica: classificazione delle parti del discorso e dei rispettivi tratti morfologici

- analisi sintattica: classificazione di alcuni ruoli sintattici e ricostruzione delle relazioni di dipendenza

- riconoscimento e classificazione semantica delle entità nominate: luoghi, persone, unità militari, aerei e navi.

Al termine delle fasi di analisi precedentemente citate, i bollettini si presentavano in formato CoNLL standard (v. fig. 3). Ogni riga del formato ConLL rappresenta un token e la sua analisi.

```
<doc url="http://www.ilc.cnr.it/w2m/doc152.html" index="152" day="131" date="1 ottobre 1915" firma="CADORNA">

1       Nella   in      E       EA      num=s|gen=f     12      comp    0       0
2       parte   parte   S       S       num=s|gen=f     1       prep    0       0
3       montuosa        montuoso        A       A       num=s|gen=f     2       mod     0       0
4       del     di      E       EA      num=s|gen=m     2       comp    0       0
5       teatro  teatro  S       S       num=s|gen=m     4       prep    0       0
6       di      di      E       E       _       5       comp    0       0
7       operazioni      operazione      S       S       num=p|gen=f     6       prep    0       0
8       nebbie  nebbia  S       S       num=p|gen=f     7       mod     0       0
9       frequenti       frequente       A       A       num=p|gen=n     8       mod     0       0
10      ed      e       C       CC      _       9       con     0       0
11      intense intenso A       A       num=p|gen=f     9       conj    0       0
12      ostacolano      ostacolare      V       V       num=p|per=3|mod=i|ten=p 0       ROOT    0       0
13      l'      il      R       RD      num=s|gen=n     14      det     0       0
14      azione  azione  S       S       num=s|gen=f     12      obj     0       0
15      delle   di      E       EA      num=p|gen=f     14      comp    0       0
16      artiglierie     artiglieria     S       S       num=p|gen=f     15      prep    0       0
17      ;       ;       F       FC      _       12      punc    0       0
```

Figura 3. Testo analizzato in formato CoNLL

Nello specifico la prima colonna di Figura 3 rappresenta l'Id ovvero un indice identificativo del token all'interno della frase, poi troviamo la forma (token) e il lemma. Seguono tre colonne relative all'analisi morfosintattica di cui le prime due corrispondono a due diversi livelli di categorizzazione di *Part Of Speech* (POS) Tagging, mentre l'ultima indica i tratti morfologici della forma. Più oltre troviamo due colonne relative all'analisi sintattica: nella prima vengono ricostruite le relazioni di dipendenza tra token assumendo che ogni dipendente abbia sempre una e una sola testa. Per ogni token viene indicato l'indice del token da cui è retto ("0" se l'elemento corrisponde alla radice della frase perciò non ha una testa), la colonna successiva ci indica se l'elemento è radice ("ROOT") e altre relazioni sintattiche. La nona colonna

indica la presenza o meno di Entità Nominate nel formato standard *IOB-format* descritto nel capitolo 2.2.3.

3.4 Luoghi

Tra le entità nominate è importante porre la nostra attenzione sui nomi di luogo in quanto punto di partenza per il successivo sviluppo del progetto. I nomi sono stati annotati manualmente sul testo e in seconda fase estratti, georeferenziati e proiettati su mappe geografiche digitali.

I toponimi dei bollettini della Prima e della Seconda Guerra Mondiale presentano caratteristiche diverse. I bollettini della Prima Guerra Mondiale, nonostante indichino per lo più luoghi italiani spesso presentano un alto numero di varianti ortografiche, ad esempio *Valle Strino*, *Val di Strino*, *Valle di Strino*, *Val Strino*, anche con diversa resa delle maiuscole *altopiano d'Asiago* e *Altopiano di Asiago*. Oppure ancora presentano casi di polimorfia toponomastica ovvero nomi diversi per indicare lo stesso luogo come *Altipiano dei Sette Comuni* e *Altopiano d'Asiago*.

Un'altra caratteristica è la specificità dei luoghi legata al tipo di combattimento che veniva attuato, metro per metro, attraverso la trincea. Compaiono nomi di massi, frazioni, vallate, speroni rocciosi, punte o cime minori sulle Alpi, fortificazioni, punti di ricovero, malghe e baraccamenti che risultano ora in larga parte distrutti. Alcuni esempi sono *Cresta Villacorna*, *Ospizio di San Bartolomeo*, *Malga Zuriz*. Nei nomi stranieri spesso compaiono errori ortografici.

Per quanto riguarda i toponimi della Seconda Guerra Mondiale, questi sono molto meno specifici, si tratta quasi sempre di città, monti, fiumi o zone più vaste, e nella maggior parte dei casi, non si tratta di nomi italiani ma della zona dell'Africa Settentrionale, Africa Orientale, Malta e zona Greco-Albanese.

Spesso questi nomi venivano trascritti con varianti ortografiche molto lontane dall'originale resa grafica, in una sorta di traduzione italianizzata della pronuncia locale

come *Adueina* al posto di *Oodweyne* in Somalia o *Arbajahn* per *Arbajahan Dam* in Kenya.

La georeferenziazione spesso non è stata semplice, a causa delle caratteristiche descritte precedentemente e delle ambiguità per omonimia dei toponimi.

Un altro problema che abbiamo dovuto affrontare riguarda il fatto che molti dei luoghi menzionati oggi non esistono più. E stato quindi necessario risalire alla posizione corretta triangolandoli con indicazioni geografiche presenti nel bollettino stesso o sfruttando vecchie carte del periodo, spesso difficili da leggere e mal digitalizzate.

Allo stesso modo abbiamo dovuto tener di conto dei cambiamenti di natura geo-politica, e di riflesso anche linguistica, che alcuni di questi nomi hanno subito nel corso del tempo, come ad esempio alcuni luoghi della regione dell'Istria che si trovavano nel territorio italiano prima della Prima Guerra Mondiale e ora si trovano in territorio sloveno o croato con nome sloveno o croato, ad esempio *Abbazia* in Croazia è ora *Opatija*.

4. Fasi di Analisi

Il progetto è costituito da cinque script in linguaggio Python, i primi quattro da applicare in sequenza sul corpus, pre-analizzato e presentato in formato CoNLL come descritto nel capitolo 3, il quinto è una possibile alternativa all'ultimo script.

Applicando gli studi relativi all'Information Extraction abbiamo cercato di estrarre gli eventi bellici, descritti da termini di dominio, e le entità nominate di tipo luogo. L'obiettivo finale è collegare correttamente ogni evento al luogo in cui si è svolto.

Prima di procedere, sono stati valutati i possibili approcci. L'estrazione di eventi bellici e relazioni attuata in modo automatico tramite modelli statistici si è resa subito una strada non percorribile a causa della peculiarità del testo stesso: la combinazione tra il dominio specifico e la lingua lontana dallo standard contemporaneo rende i bollettini di guerra un testo unico nel suo genere per la lingua italiana. Per utilizzare algoritmi di apprendimento automatico ed estrarre un modello statistico da applicare ai testi avremmo dovuto avere un training corpus di grandi dimensioni che rispecchiasse le caratteristiche dei bollettini, ma un corpus simile non esiste. Abbiamo quindi optato per un procedimento basato su regole definite manualmente e ricavate dall'analisi del testo.

4.1 Costruzione dell'Ontologia

In una prima fase è stato creato un file esterno con estensione ".txt" contenente termini che rappresentano eventi bellici.

> "[An Ontology is] A formal, explicit specification of a shared conceptualization" [15]

Un'ontologia viene definita come un modello astratto che rappresenta la conoscenza relativa a un dominio, condivisa all'interno di una comunità di esperti, comprensibile all'elaboratore e definita in modo chiaro e accessibile.

---

[15] Gruber (1993)

Il file contiene una rappresentazione organizzata di termini, e, nonostante non sia prevista un'organizzazione tassonomica dei concetti né la presenza di altri tipi di relazioni tipicamente presenti in un'ontologia, rientra nella definizione di questo termine, anche se non in senso stretto.

L'ontologia (v. Appendice per il file completo) è stata creata inizialmente con lemmi di uso più frequente ed è stata ampliata manualmente durante le fasi successive di elaborazione del progetto in modo che fosse più vicina possibile alle situazioni evidenziate nei testi in questione (allineamento ontologia - testo). Tramite lo studio dei risultati e l'osservazione diretta dei bollettini abbiamo rilevato termini ed espressioni ricorrenti che potevano essere candidati utili, e dopo attenta valutazione sono stati aggiunti o scartati.

I termini, proposti sotto forma di lemma e raggruppati per radice semantica o, in alcuni casi, per significato, sono sostantivi, verbi, nomi deverbali espressioni polirematiche o multiword composte al massimo da 4 token concatenati con underscore "_".

L'utilizzo dei lemmi consente di avere uno script molto più snello, evitando di dover ricostruire tutte le forme che una parola può assumere a causa della componente flessiva di cui è molto ricca la morfologia italiana.

Ogni riga dell'ontologia rappresenta un gruppo di termini, il primo elemento è il termine che vogliamo che sia proiettato successivamente, di solito il verbo, gli altri seguono separati con il carattere di escape "\t" o tabulazione.

4.1.1 Difficoltà

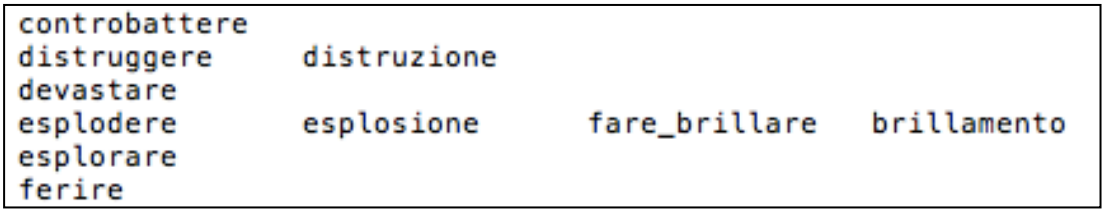
Nella fase di costruzione e popolamento dell'ontologia i problemi riscontrati sono stati molti e di diversa natura.

Il linguaggio naturale è un prodotto così ricco di sfaccettature, varietà, possibili ambiguità al punto che riuscire a definire qualsiasi termine, elemento o relazione in modo univoco e accettato da tutti, che siano esperti del settore o comuni parlanti nativi, è quasi impossibile.

Risposte a domande come “Il termine rappresenta un evento bellico?” ed “è utile ricercarlo sul testo?” non sono per nulla semplici e univoche. Da un punto di vista meramente semantico non è semplice definire se un termine debba essere considerato esplicativo di un evento bellico. Alcuni termini se decontestualizzati non sembrano avere la stessa rilevanza nel determinare la presenza di un evento (*molestare*), alcuni momenti dell’azione bellica che vengono descritti possono essere considerati marginali (*avvistamento*), effetti dell’evento (“..con concentramenti di fuoco tenemmo in vivo allarme l’avversario..” in cui l’evento è il *concentramento di fuoco* e *allarmare* risulta esserne un effetto non determinante), azioni accessorie (”La nostra aviazione ha continuato con mirabile spirito a prodigarsi..”). La decisione di non inserire tutti i termini che potessero determinare, anche vagamente, un evento o una sua parte è stata presa per selezionare gli eventi considerati più rilevanti e mantenere la qualità nel risultato finale. Il numero di eventi bellici individuati resta molto alto, ma aggiungere i termini “border line” (*provocare allarmi, aprire breccie*[16]) avrebbe creato, nella fase finale del progetto, relazioni meno significative e, in alcuni casi, impedito di trovare quelle fondamentali.

Nel caso delle espressioni poliremathiche si è reso necessario invertire l’ordine dei termini per individuare l’evento sia in una frase come “la bomba lasciata cadere…” che in “abbiamo lasciato cadere bombe…”. Un altro problema riscontrato è la presenza, all’interno delle multiword di token che non fanno parte dell’evento. Un esempio: “I nostri dirigibili eseguivano incursioni in territorio nemico, lanciando, pare con efficacia, numerose bombe...”. La maggior parte di questi casi sono stati risolti grazie all’inserimento nell’ontologia dell’ultimo termine, in particolare con “fuga”, che rappresenta effettivamente un evento, e “bomba” che non è un evento in sé ma è un termine particolarmente predittivo della presenza di un evento e tutte le volte in cui ricorre fa parte di un momento bellico da registrare (lanciare bombe, far cadere bombe, far brillare bombe, bombe esplose….). In questi casi, in cui l’ultimo termine dell’espressione è stato inserito anche come termine libero (“mettere_in_fuga” e “fuga”), in fase di matching viene data la precedenza alla multiword.

[16] Si noti il plurale “breccie” che differisce dallo standard contemporaneo “brecce”.



Cornice 1: Un'estratto dell'ontologia

La complessità del problema è dovuta anche all'alto tasso di variabilità nella realizzazione linguistica dell'evento nel testo. Oltre all'uso di un ricco vocabolario di termini bellici, è possibile individuare frasi in cui l'azione bellica non è demandata ad un singolo termine ma all'intera frase.

Infine un'ulteriore problema è dato dalla polisemia dei lemmi: un termine come "cadere" può essere esplicativo di un momento bellico come nella frase "Il velivolo cadde in un precipizio.." mentre non lo è nel caso di "…sono cadute abbondanti nevicate".

4.2 Moduli Python

Per la realizzazione degli script sono stati utilizzati diversi moduli Python. Ognuno di questi fornisce possibilità di elaborazione aggiuntive e semplifica il codice aumentandone la leggibilità. In seguito un breve riepilogo dei moduli implementati:

- *Os:* contiene molte funzioni utili per manipolare file e relativi percorsi, utile per la portabilità e lo scambio degli script.
- *Sys:* fornisce l'accesso ad alcune variabili usate o mantenute dall'interprete, e a funzioni che interagiscono con l'interprete stesso
- *Getopt:* consente agli script di analizzare gli argomenti da riga di comando
- *String:* definisce funzioni utili da applicare a variabili di tipo stringa
- *Itemgetter From Operator:* Operator consente di utilizzare un insieme di funzioni implementate in "C" che corrispondono agli operatori intrinseci di Python
- *Math:* fornisce l'accesso ad alcune funzioni matematiche.
- *Generators from future.*

4.3 Estrazione e Proiezione di Eventi Bellici

Il primo script "event_extraction.py" consente di proiettare l'ontologia sul file CoNLL.

4.3.1 Preparazione dei candidati

Nella prima parte dello script vengono preparati i candidati da proiettare come evento bellico.

Si accede all'ontologia tramite l'istruzione `with open ("Ontologia") as event_file`, in questo modo il file momentaneamente non viene aperto o letto, il che richiederebbe all'interprete Python costi superiori in termini di memoria e tempo.

Segue l'inizializzazione dei dizionari: "*event_dict*" sarà composto da coppie chiave-valore in cui la chiave è il primo elemento di ogni riga dell'ontologia e i valori sono liste contenenti i successivi elementi che appartengono alla stessa riga, qualora ci siano, altrimenti una lista vuota; "*inst_dict*" è il dizionario invertito, ogni possibile evento è chiave, il valore corrispondente è il lemma che vogliamo proiettare.

```
event_dict ={u'cadere':[u'caduta'], u'battaglia': [],
    u'cannoneggiare': [u'cannoneggiamento',u'cannonata'],

inst_dict = {u'caduta': u'cadere', u'lanciare_bomba':
    u'bombardare',
```

Cornice 2: esempio di composizione dei dizionari

In una seconda elaborazione dello script per ogni multiword sono state generate automaticamente tutte le possibili sequenze (mantenendone la lunghezza) e aggiunte alle liste dei candidati. Una espressione come "gettare_bomba" viene invertita in "bomba_gettare" (v. cap. 4.1.1)

```
13   def combinations(items, n):
14   if n==0: yield []
15   else:
16         for i in xrange(len(items)):
17             for cc in combinations(items[:i]+items[i+1:],n-
1):   18                   yield [items[i]]+cc
19   def permutations(items):
20         return combinations(items, len(items))
```

Cornice 3: creazione delle combinazioni

4.3.2 Creazione e uso della finestra scorrevole

Una volta creati i candidati e popolati i dizionari si accede al file CoNLL. Viene creata una finestra di testo scorrevole la cui dimensione è impostata a 5 per poter verificare la presenza di multiword. La finestra è più ampia della lunghezza massima in termini di token delle multiword (4 token), per evitare che termini come "*sorpresa*" o "*impresa*" fossero proiettati come "*resa*".[17]

Scorrendo le righe vengono concatenati i lemmi con il carattere *underscore* (variabile *"hash_event"*) fino al raggiungimento della dimensione della finestra, poi ad ogni lemma che viene aggiunto viene eliminato il primo della serie mantenendo la finestra a dimensione costante finché non si raggiunge la fine della frase.

Se la riga è vuota o contiene la marca xml che delimita il bollettino, la variabile *"hash_event"* viene azzerata. Nel contempo viene inizializzata e aggiornata la variabile "*candidate_event"*, una stringa che concatena "hash_event" con una sequenza di simboli fittizi ("###") per delimitare la stringa in modo inequivocabile e favorire la procedura di matching.

[17] Come si vedrà nel capitolo 4.3.3 il match viene verificato solo sulla parte terminale della stringa candidata. Ampliando la finestra, la stringa sarà delimitata a sinistra dal carattere *underscore*

4.3.3 Matching

```
93    if candidate_event.endswith("_"+a+"###"):
94          len_of_event = len(a.split("_")
95          tmp_counter = len_of_event
96          tmp_line = line_idx
97          while(tmp_counter!=0):
98                idx_dict[tmp_line] = event
99                tmp_counter-=1
99                tmp_line-=1
```

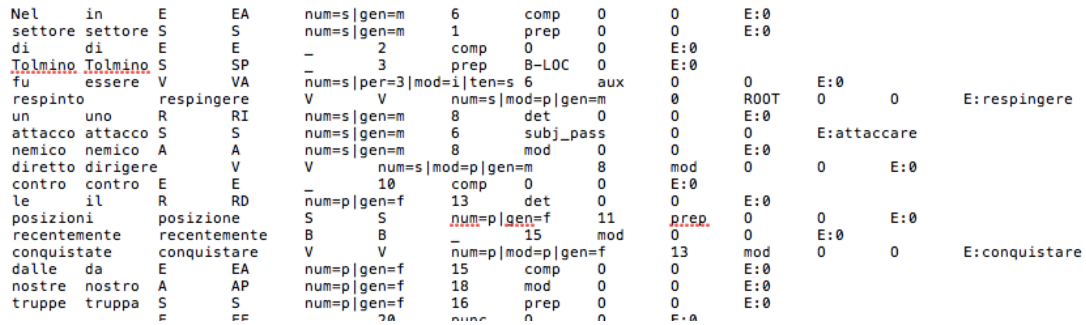
Cornice 4: Verifica del candidato e popolamento del dizionario finale degli indici

Scorrendo la lista di tutti gli eventi, se la stringa "candidate_event" termina con la sequenza underscore, evento, triplo cancelletto (vedi Cornice 4, riga 93) allora viene identificato un evento. Con il metodo `has_key(stringa)` invocato sui dizionari costruiti all'inizio dello script otteniamo il lemma da proiettare.

Se abbiamo ottenuto una corrispondenza è fondamentale definire da quanti token è composto l'elemento e impostare un contatore per proiettare l'evento designato per ogni token che faccia parte dell'evento bellico. Il dizionario *"idx_dict"* contiene l'indice e il corrispondente evento.

4.3.4 Proiezione dei candidati

Questa è l'ultima fase dello script: scorrendo nuovamente il file CoNLL per ogni riga viene verificata la presenza dell'indice nel dizionario *"idx_dict"* e creata la riga di output. Viene creata una colonna specifica che indichi la presenza o meno dell'evento bellico: "E:0" nel caso non sia un evento, altrimenti "E: LEMMA DESIGNATO"



Nel	in	E	EA	num=s|gen=m	6	comp	0	0	E:0			
settore	settore	S	S	num=s|gen=m	1	prep	0	0	E:0			
di	di	E	E	_	2	comp	0	0	E:0			
Tolmino	Tolmino	S	SP	_	3	prep	B-LOC	0	E:0			
fu	essere	V	VA	num=s|per=3|mod=i|ten=s	6	aux	0	0	E:0			
respinto		respingere		V	V	num=s|mod=p|gen=m		0	ROOT	0	0	E:respingere
un	uno	R	RI	num=s|gen=m	8	det	0	0	E:0			
attacco	attacco	S	S	num=s|gen=m	6	subj_pass		0	0	E:attaccare		
nemico	nemico	A	A	num=s|gen=m	8	mod	0	0	E:0			
diretto	dirigere		V	V	num=s|mod=p|gen=m		8	mod	0	0	E:0	
contro	contro	E	E	_	10	comp	0	0	E:0			
le	il	R	RD	num=p|gen=f	13	det	0	0	E:0			
posizioni		posizione		S	S	num=p|gen=f	11	prep	0	0	E:0	
recentemente		recentemente		B	B	_	15	mod	0	0	E:0	
conquistate		conquistare		V	V	num=p|mod=p|gen=f		13	mod	0	0	E:conquistare
dalle	da	E	EA	num=p|gen=f	15	comp	0	0	E:0			
nostre	nostro	A	AP	num=p|gen=f	18	mod	0	0	E:0			
truppe	truppa	S	S	num=p|gen=f	16	prep	0	0	E:0			

Figura 4: Una parte dell'output dello script "event_extraction.py"

4.4 Estrazione e Proiezione di NE

In questa fase vengono recuperate le Entità Nominate, già identificate in formato IOB, ricostruite se composte da più token e proiettate sul file in formato CoNLL.

4.4.1 Estrazione NE

Lo script "extract_entities.py" scorre l'output del file "event_projection.py" e compie una verifica sulla terz'ultima colonna, in cui si trovano i tag in formato IOB relativi alle entità nominate (indipendentemente dal tipo che verrà controllato in seguito). Filtrando i tag vengono ricostruite le NE composte da più token. Vengono utilizzate le forme e non il lemma per evitare spiacevoli situazioni in cui l'Entità Nominata o una sua parte sia stata elaborata in fase di lemmatizzazione ("Porto_Corsini" in "porgere_Corsini")

In conclusione lo script genera due file:

- "gg.events.entities.txt": contiene coppie (tipo entità, Entità Nominata)[18]
- "gg.entities.indexes.txt": contiene (indice, tipo di entità, Entità Nominata)

4.4.2 Proiezione NE

Lo script "ner_projection.py" viene utilizzato x proiettare le entità ricostruite nel passaggio precedente. I file passati in input saranno l'output di "event_projection.py" e il

[18] Il file è stato utilizzato per altri scopi esterni al progetto trattato.

file degli indici “gg.entities.indexes.txt”.

```
18    key_idx = int(spl_line[0])
19    elem = spl_line[-1]
20    len_of_ner = len(elem.split("_"))
21    if len_of_ner == 1:
22          entity_dict[key_idx] = elem
23    elif (len_of_ner > 0):
24          while (len_of_ner != 0):
25                entity_dict[key_idx] = elem
26                len_of_ner -= 1
27                key_idx -=1
```

Cornice 5: Una parte del codice di “ner projection.py”: controllo della lunghezza dell’Entità e popolamento del dizionario

Per proiettare correttamente l’Entità Nominata ne viene controllata la lunghezza in termini di token tramite il metodo `split("_")`[19] che trasforma la stringa in una lista di token e il metodo `len` che ne conta la lunghezza.

Il dizionario entity_dict contiene coppie chiave-valore in cui la chiave è l’indice della NE (trasformato in intero con il metodo `int`) e il valore è la NE stessa. Se la lunghezza della NE è 1, questa viene inserita direttamente nel dizionario, altrimenti tramite un ciclo `while` viene inserita la coppia nel dizionario, decrementata la lunghezza della NE e dell’indice della NE finché la lunghezza non arriva a 0. In questo modo per ogni token appartenente alla NE avremo una coppia nel dizionario.

Scorrendo successivamente il file CoNLL l’indice di ogni riga viene confrontato con le chiavi presenti nel dizionario: se la chiave è assente viene aggiunto alla stringa di output, separato da carattere di tabulazione, “0”, altrimenti viene proiettata la corrispondente Entità.

[19] Le NE formate da più token sono concatenate tramite *underscore*, per questo viene utilizzato come delimitatore.

4.5 Definizione delle relazioni Evento-Luogo

L'ultima fase prevede l'identificazione delle relazioni tra evento e luogo. L'assunto che abbiamo adottato è che l'evento e il corrispondente luogo si trovino all'interno della stessa frase. La Tabella 1 mostra tutte le possibili combinazioni tra luoghi e eventi in base al numero di entità estratte: se in una frase abbiamo un solo luogo e un solo evento la relazione sarà automaticamente creata tra le due entità, se una delle due è assente non viene identificata alcuna relazione. Le operazioni da eseguire nel caso di combinazioni "1 a molti" sono state definite in base ai testi stessi calcolando quale eventualità portasse ad un migliore risultato. Il caso "molti a molti" è stato trattato seguendo due approcci diversi: il primo prevede la definizione della relazione per adiacenza, il secondo per dipendenza sintattica.

Numero Luoghi	**Numero Eventi**	**Esito**
0	0, 1, Molti	Non c'è relazione
1, Molti	0	Non c'è relazione
1	1	Relazione L1- E1
1	Molti	Per ogni E viene creata una relazione con il L
Molti	1	L'E è in relazione con ogni L
Molti	Molti	Relazione per adiacenza o per dipendenza sintattica

Tab 1: Le possibili combinazioni tra il numero di luoghi e di eventi e le procedure da applicare per determinare le relazioni.

4.5.1. Relazioni per Adiacenza

Il primo tentativo per la definizione delle relazioni evento-luogo è basato sull'adiacenza tra due entità. Non avendo la possibilità di applicare algoritmi di apprendimento

automatico per l'estrazione di feature o pattern per il riconoscimento della struttura della frase abbiamo dovuto applicare metodi basati su regole. L'assunto su cui si basa lo script è che ogni luogo si riferisca all'evento che si trova linearmente più vicino, all'interno di una finestra di contesto equivalente alla dimensione della frase che stiamo analizzando.

Lo script "*links_ev_loc.py*" prevede due funzioni, la prima per la creazione delle relazioni e la seconda per la proiezione delle stesse.

Scorrendo le righe del file in input (corrispondente al file output di "ner_projection.py") viene isolata la frase creando variabili temporanee (relative alla frase): due set, uno per i luoghi, l'altro per gli eventi bellici e un dizionario.

Se la riga inizia con un carattere numerico[20], ovvero l'indice, ne vengono separati gli elementi e controllate le colonne proiettate con i due precedenti script e quella relativa al tipo di entità.

Con lo script "ner_projection.py" avevamo proiettato tutti i tipi di NE, se troviamo una entità, `loc_col != "0"`,ne viene controllato il tipo: se si tratta di un luogo viene inserito nel set `tmp_loc_set` e aggiunto al dizionario `dict_locx_temp` in cui la chiave è l'indice della NE e come valori inseriamo una lista con il nome del luogo, un indice per il conteggio della distanza tra luogo e evento inizializzato a 500 e un indice inizializzato a 0 che viene aggiornato successivamente con l'indice dell'evento a cui il luogo dovrà essere collegato.

```
dict_locx_temp[token_id] = [loc_col, 500, 0]
```

Cornice 6: Popolamento del dizionario temporaneo

Se troviamo un evento, ne aggiungiamo l'indice al set `tmp_events`.

La struttura del testo in input prevede la marca xml *doc* di apertura, una riga vuota, il testo della frase in formato CoNLL, una riga vuota come separatore di fine frase e la riga contenente la marca xml *doc* di chiusura oppure l'inizio di un'altra frase. Quando raggiungiamo una riga vuota possiamo trovarci a fine frase oppure all'inizio della frase,

[20] Tutte le righe escluse quelle contenenti le marche xml e le righe vuote.

dopo la marca xml. I set e il dizionario creati precedentemente sono temporanei, relativi alla frase e vengono inizializzati tutte le volte che troviamo una riga vuota, non prima di aver compiuto alcune operazioni.

Se la riga è vuota e i due set `tmp_events` e `tmp_loc_set` contengono un numero di elementi superiore a 1, otterremo una frase con rapporto "molti a molti". Per definire tra quali entità creare la relazione viene conteggiata la distanza tra il primo luogo e ogni evento. Ogni conteggio, in valore assoluto (vedi Cornice 7, linea 31) viene confrontato con il valore indicante la distanza precedentemente calcolata (se è la prima volta sarà 500, come da inizializzazione); se il nuovo valore è inferiore significa che abbiamo trovato un nuovo evento candidato a determinare la relazione con il luogo, in questo caso si procede sostituendo il valore della distanza `count` con quello nuovo `tmp_count` e l'indice dell'evento candidato `idx_ev_count` (inizialmente 0) verrà aggiornato con l'indice dell'evento candidato. Si procede iterativamente fino ad aver esaurito tutti gli eventi presenti nel set.

A questo punto abbiamo ottenuto l'evento con minima distanza dal luogo preso in considerazione. Prima di procedere con il calcolo delle relazioni che riguardano gli altri luoghi inseriamo la coppia appena individuata nel dizionario `dict_idx`, la chiave sarà l'indice dell'evento `(idx_ev_count)` e il valore il nome del luogo `(val_loc[0])`.

Esiste la possibilità, ed è anzi probabile, che nel dizionario sia già presente la chiave che noi vogliamo inserire perché un evento partecipa anche ad un'altra o più relazioni. Per definizione le chiavi di un dizionario sono univoche e non è possibile avere più coppie con la stessa chiave. Per ovviare al problema, se la chiave è già presente, viene registrato il vecchio valore in una variabile temporanea, vi viene aggiunto il nuovo valore creando una lista, infine viene sovrascritta la coppia con quella nuova.

Se la frase non è del tipo "molti a molti", per ogni evento (può essere anche uno solo) viene aggiunta al dizionario `dict_idx` una coppia "indice evento – lista luoghi", a meno che uno dei due set non sia vuoto. In questo caso non esiste relazione. Prima di passare alla frase successiva vengono azzerati i set e il dizionario `dict_locx_tmp` .La

proiezione delle relazioni avviene in modo simile alla proiezione di eventi e NE verificando la presenza dell'indice nel dizionario `dict_idx`.

```
26    if len(tmp_events) >1 and len(tmp_loc_set) > 1:
27          for idx_loc, val_loc in
dict_locx_temp.iteritems():
28                count = val_loc[1]
29                idx_ev_count = val_loc[2]
30                for idx_ev in tmp_events:
31                      tmp_count = math.fabs(idx_loc - idx_ev)
32                      if tmp_count < count:
33                            count = tmp_count
34                     idx_ev_count = idx_ev
```

Cornice 7: Calcolo dell'evento con distanza minima dal luogo

In output viene creata una nuova colonna, la relazione viene proiettata sulla riga dell'evento nel formato "EVENTO: Luogo" o "EVENTO: Luogo1,Luogo2", altrimenti "0".

```
Nel     in      E       EA    O       O       E:0     0       0
settore settore S       S     O       E:0     0       0
di      di      E       E     O       E:0     0       0
Monfalcone      Monfalcone      B-LOC   O       E:0     Monfalcone      0
,       ,       F       FF    E:0     0       0
dopo    dopo    E       E     O       E:0     0       0
due     due     N       N     E:0     0       0
giorni  giorno  S       S     O       E:0     0       0
di      di      E       E     E:0     0       0
accanito        accanito        mod     O       O       E:0     0       0
combattimento   combattimento   prep    O       O       E:combattere    0       COMBATTERE:Monfalcone
,       ,       F       FF    E:0     0       0
conquistarono   conquistare     0       ROOT    O       O       E:conquistare   0       CONQUISTARE:Quota_121,Debeli
le      il      R       RD    O       E:0     0       0
alture  altura  S       S     O       E:0     0       0
di      di      E       E     E:0     0       0
Quota   Quota   S       SP    E:0     Quota_121       0
121     121     N       N     E:0     Quota_121       0
e       e       C       CC    E:0     0       0
del     di      E       EA    O       E:0     0       0
Debeli  Debeli  S       SP    E:0     Debeli  0
.       .       F       FS    E:0     0       0
```

Figura 5: Proiezione delle relazioni in base all'adiacenza. (Le colonne centrali del formato CoNLL sono state tagliate per migliorare la visualizzazione del risultato finale)

4.5.2. Relazioni per Dipendenza Sintattica

Il secondo tentativo per la definizione delle relazioni evento-luogo è basato sulla dipendenza sintattica tra token all'interno della frase. Scorrendo le relazioni di dipendenza sintattica, indicate nella settima colonna del formato CoNLL del testo in input (v. cap. 3.3 e Figura 3), si risale dall'Entità Nominata di tipo luogo all'evento bellico identificato precedentemente.

L'assunto su cui si basa lo script è che ogni luogo sia retto sintatticamente da un evento. A causa della non totale veridicità dell'assunto e della difficoltà del parser ad adeguarsi ad un testo sintatticamente così lontano dallo standard contemporaneo, in molte occasioni l'algoritmo non giunge alla definizione di una relazione. Non potendo procedere ulteriormente con il parser sintattico, in queste situazioni, l'algoritmo procede con la ricerca lineare per adiacenza.

Come il precedente, lo script "*sint_links.py*" prevede due funzioni, la prima per la creazione delle relazioni e la seconda per la proiezione delle stesse. Concettualmente lo schema di base è lo stesso di quello per adiacenza: viene isolata la frase e vengono creati set e dizionari per avere a disposizione solo le informazioni utili per i calcoli successivi. Una volta raggiunta la riga vuota e controllato che sia il separatore tra frasi e non la riga compresa tra la marca xml di apertura del bollettino e il primo token della frase, se la frase presenta un rapporto tra il numero di NE di luogo e il numero di eventi bellici di "molti a molti" si va a ricercare tra quali entità ci sia una relazione, altrimenti per ogni evento creiamo una coppia nel dizionario finale `dict_idx` nel formato "indice dell'evento-NE" (oppure lista di NE se sono più di una). Infine vengono confrontati gli indici e proiettate le relazioni come nello script "*links_ev_loc.py*"

La colonna del formato ConLL relativa alla dipendenza sintattica riporta per ogni token l'indice del token da cui è retto sintatticamente, "0" se stiamo osservando la radice. La definizione della relazione evento-luogo viene determinata in base alla dipendenza sintattica.

Per ogni frase vengono registrate le informazioni necessarie in un dizionario di dizionari `sentences_dict`.

```
sentences_dict[sentences_counter][token_in_sentence] =
[loc_col, ev_col, ner_col, sint_col, token_id, root]
```

Cornice 8: Il dizionario di dizionari

La chiave `sentences_counter` è un contatore, inizializzato a 1 e indica il numero della frase. La chiave del dizionario interno è `token_in_sentence`, l'indice relativo alla frase del token che stiamo osservando. I valori che vengono inseriti (vedi cornice 8) sono, in ordine, la NE luogo, la proiezione dell'evento, il tipo dell'entità, l'indice relativo alla dipendenza sintattica, l'indice totale del token, l'individuazione della radice.

Scorrendo le righe del file in input (corrispondente al file output di "ner_projection.py") viene isolata la frase creando variabili temporanee: tre set di cui uno per gli eventi bellici `tmp_events` con gli indici[21] degli eventi e due per i luoghi ovvero `tmp_loc_set` contenente i nomi di luogo e `tmp_loc_idx` con gli indici[22] dei luoghi.

Una volta raggiunta la riga vuota, viene controllata la lunghezza di `sentences_dict[sentences_counter];` se è maggiore di 0 significa che la riga funge da separatore di frase, e non è la riga compresa tra la marca xml di apertura del bollettino e il primo token. Viene verificato che la frase sia di tipo "molti a molti".

Per ogni luogo, per determinare con quale evento formi una relazione, si osserva, all'interno del dizionario `sentences_dict[sentences_counter],` l'indice riportato nella colonna sintattica. Per verificare se l'indice corrisponde ad un evento, questo viene cercato come chiave nel dizionario stesso. La ricerca procede iterativamente finché non si raggiunge una riga contenente un evento bellico o la radice.

Nel caso che con l'iterazione sia stata raggiunta la radice e non un evento bellico, le motivazioni possono essere principalmente due:

- il parser sintattico ha commesso degli errori a causa della particolarità sintattica dei testi in esame

[21] Vengono utilizzati gli indici dell'intero file.
[22] In questo caso vengono utilizzati gli indici interni alla frase.

- non sussiste una relazione tra il luogo e alcun evento bellico presente nella stessa frase.

La seconda possibilità si pone in antitesi con l'assunto di partenza, ovvero che dato un luogo e uno o più eventi bellici all'interno della stessa frase esiste sempre una relazione tra il luogo e almeno uno degli eventi.

Per ovviare al problema, nel caso la ricerca su base sintattica non giunga a buon fine, nello script è stata incorporata una parte del codice presentato nel capitolo 4.5.1. e la ricerca procede con il metodo di adiacenza.

Il formato di output per i due script che determinano le relazioni è lo stesso: viene creata una colonna nuova e sulla riga degli eventi che formano una relazione verrà proiettato "EVENTO: Luogo" o "EVENTO: Luogo1,Luogo2", altrimenti "0".

5. Valutazione delle fasi di Analisi

In questo capitolo verranno presentati i risultati ottenuti nelle diverse fasi di analisi in termini di *precision*, *recall*, e F-measure, e le metodologie di valutazione applicate.

5.1 Valutazione Proiezione Eventi

La valutazione della fase di estrazione e proiezione degli eventi bellici è stata effettuata estraendo un campione casuale di 100 bollettini sia per i testi della Prima Guerra Mondiale che per quelli della Seconda. Questi sono stati annotati manualmente con i corretti tag di output creando il Corpus Gold Standard di riferimento.

La decisione di verificare gli eventi individuati frase per frase invece che token per token è dovuta alla forma degli eventi bellici e alle scelte operate nello sviluppo del sistema. In particolare la presenza di multiword e la modalità con cui abbiamo risolto il problema delle sequenze contenenti token esterni all'evento, rendeva il calcolo per token molto più complesso e penalizzante. Un'errata o mancata estrazione di una multiword, per esempio, avrebbe prodotto un numero di errori pari alla lunghezza in termini di token della sequenza stessa.

L'output dello script "event_projection.py" è stato confrontato con il Gold Standard per mezzo di procedure automatiche.

Per ognuno dei due file, per ogni frase, sono stati raccolti gli eventi proiettati sotto forma di set e inseriti in due dizionari, uno contenente i dati relativi all'output generato dall'algoritmo e l'altro relativo al Gold Standard. La chiave dei dizionari è un numero progressivo corrispondente al numero della frase, uguale per entrambi per poter allineare i valori corrispondenti. Un set è una collezione non ordinata di valori[23], una struttura dati basata sulla teoria degli insiemi e possiede metodi per verificarne le proprietà; pertanto un set A è uguale ad un altro set, B, se e solo se A è sottoinsieme di B e B è sottoinsieme di A. Tramite il metodo `issubset` viene verificato che i due set siano l'uno il sottoinsieme dell'altro e quindi equivalenti. Se la condizione si verifica il contatore degli

[23] Guido van Rossum, Manuale di riferimento Python

output corretti viene incrementato della lunghezza del set stesso (ogni true positive incrementa di 1 il contatore). Qualora i due insiemi non siano identici, viene valutato ogni elemento che è stato estratto: se presente anche nel Gold Standard viene incrementato di uno il contatore relativo ai true positive, se assente viene incrementato quello degli output errati (false positive). Infine, con il metodo `difference`[24] vengono estratti gli elementi che appartengono al Gold Standard ma non sono presenti nel nostro set e viene incrementato il contatore degli output mancati (false negative) di tante unità quanti sono gli elementi restituiti dal metodo.

Ottenuti i valori di true positive, false positive, e false negative si è proceduto con il calcolo di precision, recall e F-measure come illustrato nel capitolo 2.5

I risultati, visibili in Tabella 2, ci mostrano percentuali apprezzabili, ottime per quanto riguarda la precisione che raggiunge il 90% di correttezza, e molto soddisfacenti per quanto riguarda la copertura che supera l'80%. Nel calcolo della F-measure è stato attribuito lo stesso peso a precision e recall. Le performance ottenute sui bollettini della Seconda Guerra Mondiale sono ottime e indicano un incremento di 7 punti percentuali per la recall rispetto a quelle ottenute sui bollettini della Prima.

Risultati così alti, soprattutto per quanto riguarda i bollettini della Seconda Guerra Mondiale, si spiegano con l'abbondanza di termini che indicano eventi bellici, la loro combinazione nel descrivere un evento complesso e il livello di annotazione manuale non troppo granulare e che, nel caso di ambiguità (sia nella scelta del tipo di etichetta da proiettare, sia per l'estrazione o meno di eventi border line, v.cap. 4.1.1) è stato allineato con il risultato generato dall'algoritmo.

Nella frase "Il piccolo rimorchiatore, colpito durante il combattimento, si incendiava ed in seguito affondava" gli eventi bellici individuati sono *colpire, combattimento, incendiare* e *affondare,* 4 in soli 15 token.

In molti casi lo stesso evento viene descritto da più termini, come nella frase "… è stato colpito dal fuoco delle mitragliatrici e abbattuto…" gli eventi estratti sono *colpire* e *abbattere.* In questo caso avremmo potuto preferire l'estrazione di *mitragliare* invece

[24] Restituisce una lista di elementi ottenuti dalla differenza set A – set B

che *colpire.* Nonostante questo in fase di annotazione del Gold Standard la scelta è stata quella di avvicinarsi all'output generato automaticamente. Sia *mitragliare* che *colpire* in questo caso descrivono lo stesso evento, e l'avremmo accettato come corretto in entrambi i casi. Per il nostro obiettivo era fondamentale che fosse estratto l'evento come tale, l'etichetta che vi viene apposta (il tipo di evento estratto e proiettato) rimane secondaria in casi di ambiguità come questo.

Un così alto numero di eventi correlato con l'ambiguità nelle espressioni e, di conseguenza, nella creazione del relativo Gold Standard ha fatto si che i valori relativi a precision e recall fossero molto più alti delle aspettative.

	1° GM	**2°GM**
Precision	90.616	94.807
Recall	82.338	89.193
F-measure	86.279	91.914

Tab 2: I risultati relativi all'estrazione di eventi bellici

5.2 Valutazione NER

Il compito di Named Entity Recognition era già stato portato avanti in modo automatico in fasi di elaborazione precedenti alla realizzazione di questo progetto. Per completezza vengono qui riportate le valutazioni relative al riconoscimento di nomi di luogo[25].
L'estrazione delle entità nominate dei testi della Prima Guerra Mondiale è stata successivamente rivista e corretta manualmente.

	1° GM	**2° GM**
F-measure	81.8	95.0

Tab 3: I risultati relativi all'estrazione di nomi di luogo

[25] Lenci & Passaro (2014)

5.3 Valutazione del Riconoscimento di Relazioni

L'annotazione del Gold Standard e, conseguentemente, la valutazione del processo è stata effettuata al netto della proiezione degli eventi bellici e delle Entità Nominate per evitare che le percentuali di errore si propagassero a cascata.
Si è proceduto annotando manualmente 100 bollettini per ogni corpus con le relazioni "evento-luogo". Per ogni frase sono state estratte coppie Evento-Luogo, se un evento presenta molteplici relazioni vengono estratte un numero di coppie pari al numero di luoghi a cui l'evento è abbinato. Le coppie vengono filtrate in modo che non vi siano duplicati.
Con un procedimento simile a quello descritto per il confronto e la valutazione della fase di estrazione di eventi, vengono creati due dizionari. La lista di tutte le coppie presenti nella frase viene inserita in un dizionario come valore, la chiave è il numero della frase. Per ogni coppia presente in entrambi i dizionari viene incrementato il contatore relativo ai true positive, per ogni coppia estratta ma non presente nel Gold Standard viene incrementato il contatore dei false positive, e, al contrario per ogni coppia non estratta ma presente nel Gold Standard viene incrementato il contatore dei false negative. Ogni incremento è pari a una unità.
Infine il calcolo di precision, recall e F-measure (v. cap. 2.5)

I risultati, visibili in tabella 4, mostrano percentuali simili tra i due corpora e un miglioramento di 3 - 4 punti percentuali in termini di precision, e di 2 punti in termini di recall a favore dell'algoritmo basato sulla dipendenza sintattica (integrato con la ricerca lineare).

Un dato importante e inatteso è quello relativo all'algoritmo basato sulla dipendenza sintattica privato della componente lineare. In questo modo si ottengono le migliori valutazioni per la precision, mentre si registra un crollo per i valori di recall, che diminuisce di 16 punti percentuali nei testi della Prima Guerra Mondiale. Valori così bassi di recall vengono in parte compensati dai valori molto alti di precision, tanto che, nei testi della Seconda Guerra Mondiale, la F-measure calcolata si discosta dal valore estratto con metodo di adiacenza solo di un punto percentuale.

	Precision	Recall	F-measure
1° GM Adiacenza	91.704	84.237	87.812
1° GM Dipendenza sint + Adiacenza	94.839	86.325	90.382
1°GM Dipendenza sintattica	95.584	70.041	80.843
2° GM Adiacenza	91.200	85.074	88.030
2° GM Dipendenza sint + Adiacenza	93.869	87.126	90.372
2°GM Dipendenza sintattica	94.347	80.970	87.148

Tab 4: I risultati relativi all'estrazione di relazioni.

Nonostante la diversa struttura sintattica dei testi dei due conflitti, i dati mostrano che quando l'algoritmo in base sintattica arriva a definire una relazione, lo fa con una correttezza superiore al 94 %.

6. Conclusioni

Prima di ripercorrere le fasi di elaborazione del progetto e valutarne i risultati, è giusto ricordare le peculiarità dei bollettini di guerra, testi unici nel loro genere per struttura, contenuto di dominio specifico e aspetti linguistici lontani dallo standard contemporaneo. L'assenza di un corpus d'addestramento per allenare gli strumenti ai vari livelli di analisi ci ha portati ad utilizzare sistemi di estrazione di informazione basati su regole, create in modo da essere il più possibile adattate ai testi in esame. Le difficoltà poste erano molteplici e siamo riusciti a risolvere la maggior parte di esse ottenendo risultati nel complesso molto soddisfacenti.

Nonostante gli ottimi risultati e le difficoltà imposte dal tipo di testo ritengo che ci siano spazi di miglioramento sia in fase di estrazione degli eventi che in fase di definizione delle relazioni Evento-Luogo. In particolare la ricerca sul testo degli eventi bellici potrebbe essere affinata apportando modifiche all'algoritmo: un esempio è l'identificazione di un evento rappresentato da una sequenza di termini che presenta all'interno lemmi che non ne fanno parte (*prendere prigionieri* e *prendere una decina di prigionieri*). Riuscendo quindi a gestire finestre contenenti numeri variabili di token potremmo aumentare la copertura del sistema. Un'altra situazione spesso riscontrata e migliorabile è la presenza di eventi seguiti o preceduti da una negazione (*senza produrre alcun danno, nessuna azione*). In questo caso l'evento viene proiettato producendo un output errato e diminuendo i valori relativi alla precision.

Per quanto riguarda l'estrazione di relazioni, i due diversi tentativi hanno restituito buoni risultati singolarmente e la versione finale basata sulla dipendenza sintattica, coadiuvata dall'adiacenza lineare quando la prima non portava ad un risultato, ha migliorato ulteriormente i valori raggiunti in termini di F-measure. Tuttavia potrebbe essere interessante, anche se molto complesso, cercare di affinare il set dei luoghi estratti mantenendo solo quelli fondamentali. Spesso, oltre al nome proprio, ci sono altre indicazioni geografiche per lo stesso luogo, come il nome della zona o regione in cui si trova, o *a nord di, 10 chilometri a sud di*. In questi casi viene creata una relazione sia tra il luogo "effettivo" che con gli altri luoghi che generalizzano o definiscono con più

precisione la posizione del primo. Così vengono create una serie di relazioni accettabili e corrette ma non ottimali né fondamentali.

In conclusione possiamo esprimere soddisfazione per le valutazioni estratte, che si attestano, in termini di F-measure, attorno all' 86% e 91%, rispettivamente per i bollettini della Prima e della Seconda Guerra Mondiale per il compito di estrazione di eventi bellici e al 90 % per quanto riguarda la definizione di relazioni, un ottimo punto di partenza per eventuali approfondimenti futuri.

7. Bibliografia

Boschetti, Federico, Andrea Cimino, Felice Dell'Orletta, Gianluca Lebani, Lucia Passaro, Paolo Picchi, Giulia Venturi, Simonetta Montemagni, Alessandro Lenci. 2014. *Computational Analysis of Historical Documents: An Application to Italian War Bulletins in World War I and II*, in Proceedings of the LREC 2014 Workshop on "Language resources and technologies for processing and linking historical documents and archives – Deploying Linked Open Data in Cultural Heritage" (LRT4HDA 2014), Reykjavik, 26 May 2014.

D'Achille,Paolo. 2010. *L'Italiano Contemporaneo*. Bologna, Il Mulino.

Gruber, T.R. 1993. *A Translation Approach to Portable Ontologies*. In "Knowledge Acquisition" 5(2):199–220, 1993.

Jackson, P. and I. Moulinier. 2002. *Natural language processing for online applications: text retrieval, extraction, and categorization.* John Benjamins Publishing Company.

Jiang, Jing. 2012. *Information Extraction from Text*. In Charu C. Aggarwal and ChengXiang Zhai (Eds.), "Mining Text Data", Springer.

Jurafsky, Daniel and James H. Martin. 2006. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Londra, Prentice-Hall.

Mitkow, Ruslan. 2004. *The Oxford Handbook of Computational Linguistics (Oxford Handbooks in Linguistics S.)*. Oxford, Oxford University Press.

Passaro, Lucia e Alessandro Lenci. 2014. *"Il Piave mormorava…": Recognizing Locations and other Named Entities in Italian Texts on the Great War,* in "Proceedings of the First Italian Conference on Computational Linguistics", Pisa, Italy, December 9-10 2014: 286–290

Piskorski, Jakub and Roman Yangarber. 2013. *Information Extraction: Past, Present and Future*. In T. Poibeau, H. Saggion, J. Piskorski, and R. Yangarber, editors, "Multi-source, Multilingual Information Extraction and Summarization". Volume in the Series: "Theory and Applications of Natural Language Processing". Springer-Verlag, Berlin & New York.

Salvi, Giampaolo e Laura Vanelli. 2004. *Nuova Grammatica Italiana*. Bologna, il Mulino.

Sang, E. F. T. K. and De Meulder, F. 2003. *Introduction to the conll-2003 shared task: Language-independent named entity recognition*. In Daelemans, W. and Os- borne, M. (Eds.), "Proceedings of CoNLL-2003", pp. 142– 147. Edmonton, Canada.

Sarawagi, Sunita. 2007. *Information Extraction*, in "Foundations and Trends in Databases", Vol. 1, No. 3.

van Rijsbergen, C. J. 1975. *Information Retrieval*. London, Butterworths.

8. Sitografia

Ali e Uomini:

http://www.alieuomini.it//homepage

Evalita: risultati 2009

http://www.evalita.it/2009/results

Memorie di Guerra:

http://www.memoriediguerra.it/wwm/

Python:

http://www.python.it

Guido Van Rossum, Manuale di riferimento Python:

http://docs.python.it/html/ref/

9. Appendice

9.1. Ontologia

In seguito viene riportata l'intera ontologia contenente i termini bellici.

abbattere	abbattuto
annientare	
abbandonare	
accerchiare	accerchiamento
affondare	affondamento
assaltare	assalto
attaccare	attacco
atterrare	atterraggio
avanzare	avanzata
bombardare	bombardamento gettare_bomba lasciare_cadere_bomba lanciare_bomba lancio_di_bomba getto_di_numeroso_bomba getto_di_bomba lanciare_qualche_bomba bomba_essere_cadere lanciare_numeroso_bomba bomba lasciare_cadere_alcuna_bomba sganciare_bomba
lanciare_granata	granata
cadere	caduta
catturare	cattura
danneggiare	causare_danno infliggere_danno recare_danno
infliggere_perdita	causare_vittima fare_vittima
subire_perdita	subire_vittima subire_danno
prendere_prigioniero	fare_prigioniero riportare_prigioniero trarre_prigioniero
lasciare_prigioniero	perdere_prigioniero abbandonare_prigioniero
fulminare	
centrare	
colare_a_picco	colare_li_a_picco
colpire	dirigere_colpo colpo_di_mano
combattere	combattimento attività_combattivo
compiere_azione	effettuare_azione
cannoneggiare	cannoneggiamento cannonata
conquistare	conquista
impadronirsi	
riconquistare	riconquista
contrattaccare	contrattacco
contrastare	
controbattere	
distruggere	distruzione
devastare	
esplodere	esplosione fare_brillare brillamento

esplorare	
ferire	
arretrare	indietreggiare
incendiare	incendio mettere_a_fuoco
intercettare	
mitragliare	mitragliamento mitragliata
molestare	tiro_di_molestia azione_di_molestia
occupare	occupazione rioccupare
precipitare	infrangere_a_suolo
respingere	ributtare
ripiegare	ripiegamento
salvare	porre_in_salvo mettere_in_salvo trarre_in_salvo
sbarcare	sbarco
scortare	
sganciare	
silurare	siluramento
sorvolare	
spezzonare	spezzonamento lanciare_spezzone lancio_di_spezzone
uccidere	
successo	
tiro_di_artiglieria	azione_di_artiglieria attività_di_artiglieria concentramento_di_artiglieria duello_di_artiglieria duellare_di_artiglieria fuoco_di_artiglieria duello_di_fuoco appoggio_di_artiglieria tiro_di_interdizione tiro_di_disturbo preparazione_di_artiglieria
vittoria	
resa	
resistere	resistenza opporre_resistenza opporre_si
ritirata ritirarsi ritirare	
offensiva	azione_offensivo attività_offensivo ritorno_offensivo operazione_offensivo
controffensiva	azione_controffensivo
operazione	
puntata	
rastrellamento	rastrellare
reazione	
sbarramento	
scontro	
scoppio	
slancio	
sortita	
ammassamento	
avvicinamento	
battaglia	

crollo	
Fuggire	mettere_in_fuga porre_in_fuga volgere_in_fuga fuga fugare
incursione	effettuare_incursione compiere_incursione eseguire_incursione avere_incursionato
infiltrazione	
inseguire	inseguimento
intervento	
lottare	lotta
martellare	martellamento
bersagliare	
espugnare	
assalireassalti	
ricognizione	
irrompere	irruzione
cacciare	scacciare ricacciare
disperdere	
sorprendere	
sventare	
ridurre_a_silenzio	
uso_di_gas	gas
lanciare_freccia	
attività_aereo	
reazione_contraereo	
azione	agire
rifugiare	
sconfiggere	sconfitta
battere	
sottomettersi	sottomissione
sbaragliare	
urtare	
speronare	
inabissare	
assediare	
saltare_in_aria	
rintuzzare	
aprire_il_fuoco	
sparare	

Ringraziamenti

Un dovuto ringranziamento va a tutti colo che mi hanno accompagnata, anche solo per un istante durante questo percorso.

In particolare vorrei ringraziare chi ci ha creduto, chi oggi ha voluto essere qui e anche chi non ha potuto.

A mia madre e mio padre.

A Franco e alla sua versione personale del World Wide Web.

A Molly, Caraffa, Lalla, Clau, Giamba, Slash, Marghe e Tere per esserci ancora, per la pazienza e la sopportazione, per le liti e i momenti magici e per almeno altri cento motivi. La persona che sono oggi vi deve davvero molto.

A Mattia che merita un ringraziamento a parte solo per essere com'è; a Nicholas, Sara, Roberto, Giulio e Guglielmo con cui esami e lezioni sono stati più leggeri.

A Michele e alla sua musica, sottofondo nella stesura di questo elaborato.

A Diana, Vale, Ceci, Gian e Fra presenze lontane ma costanti e imprescindibili.

A chef Max, Ste, Costy, Ale, Davide, Salvo, Miche, Ricca, Elida, Pietro e Mario per il tempo passato insieme ed essere diventati importanti in così poco tempo.

A tutti quelli che non ho nominato ma che hanno arricchito le mie giornate con un palloncino (o un albero di palloncini), un post-it, uno smile o un pensiero.

Grazie.