Università degli studi di Torino

Dipartimento di Informatica

Tecnologie del Linguaggio Naturale

Prima Parte

Report di progetto: Traduttore Transfer sintattico IT→EN

Studenti
Ottina Marco
Sgaramella Davide

A.A. 2019/2020

Indice

Introduzione

Per il progetto proposto si chiedeva di realizzare un Traduttore Transfer sintattico da Italiano ad Inglese.
Quindi si è sviluppato un programma che, prendendo in input una frase in italiano, fornisse in output la corrispondente in inglese. Questo avviene mediante l'utilizzo di due risorse principali:

- **Tint;**
- **SimpleNLG.**

Attraverso le componenti appena citate si sono potuti applicare i concetti visti durante il corso e, di conseguenza, "toccare con mano" il processo di elaborazione di una frase o di un testo, con tutte le sfaccettature che ne comporta.

Capitolo 1

Soluzione proposta

Come evidenziato dalla traccia ci sono alcune frasi da tradurre. Quindi come prima cosa si è cercato di fare luce sui vari step da intraprendere per lo svolgimento del progetto.
Essenzialmente sono stati individuati 3 step fondamentali:

- **Step 1**: Parsing;
- **Step 2**: Elaborazione PoS e Sentence Plan;
- **Step 3**: Traduzione e generazione della frase.

Step 1
Per prima cosa si è scelto di utilizzare **Tint** per effettuare il parsing delle frasi fornite, le quali sono:

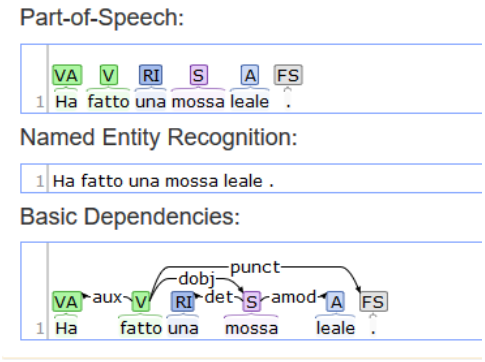
- È la spada laser di tuo padre;
- Ha fatto una mossa leale;
- Gli ultimi avanzi della vecchia Repubblica sono stati spazzati via.

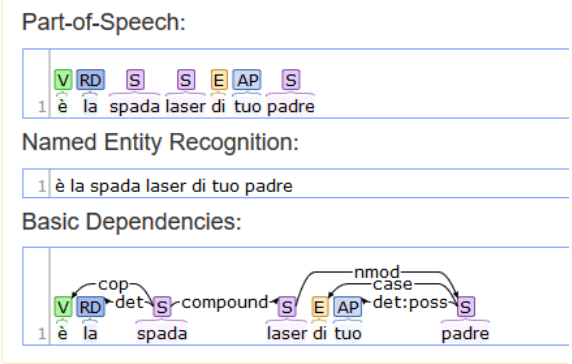
L'operazione di parsing produce un Json in cui sono annotate le dipendenze che si trovano all'interno della frase analizzata con gli opportuni riferimenti. Ad esempio, per una certa parola analizzata viene specificato il suo "index", ovvero la posizione che essa occupa nella frase. Inoltre ci sono altre informazioni importanti per il

prosieguo del progetto come la dipendenza che la parola rappresenta e soprattutto i Governor e Dependent. I suoi campi sono:

- **dep**: tipologia dipendenza.
- **governorGloss**: head della dipendenza.
- **dependentGloss**: dependent della dipendenza.

Le dipendenze vengono poi sfruttate per la costruzione della struttura sintattica ad albero su cui lavorare. Un altro aspetto molto importante riguarda le feature linguistiche fornite da Tint stesso: quelle relative alle forme verbali permetteranno poi il corretto settaggio dei metodi per l'impostazione del tempo verbale all'interno della frase.
Un altro strumento che Tint mette a disposizione è la visualizzazione grafica degli alberi generati:





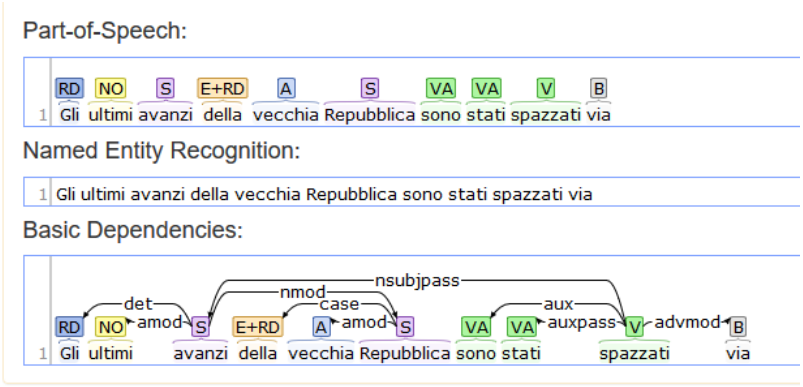


Figura 1.1: Alberi a dipendenze delle frasi analizzate.

Attraverso queste rappresentazioni è stato possibile effettuare il mapping tra le dipendenze e la struttura di SimpleNLG. Inoltre, sono state molto utili per capire come muoversi nella costruzione della frase.

Step 2

Una volta ottenute le informazioni necessarie dalla fase precedente è possibile passare alla creazione del Sentence Plan. In questo caso abbiamo utilizzato solo alcune delle info elaborate, come ad esempio il PoS, la stringa elaborata e la feature (ad esempio, per un verbo il tempo di coniugazione); a questo si aggiunge un ID andando a formare una quadrupla che identifica la parola nel Sentence Plan.

Grazie al mapping tra le dipendenze di Tint e la struttura di SimpleNLG si possono implementare dei controlli attraverso cui analizzare l'albero generato e in seguito impostare i parametri del Sentence Plan stesso.

Step 3

Una volta impostata quindi la costruzione della frase, si passa alla traduzione dei vari lessemi. Si è realizzato un piccolo dizionario in cui sono presenti le parole contenute nelle frasi, con l'aggiunta di altre affini. Effettuata quindi la traduzione, si passa alla fase di generazione e successivamente all'output.

1.1 Risorse utilizzate

1.1.1 Tint (The Italian NLP Tool)

Tale strumento è una pipeline per **Natural Language Processing (NLP)** in italiano. È molto veloce e preciso e implementa la maggior parte degli strumenti linguistici comuni, come la codifica delle parti del discorso e l'analisi delle dipendenze. Lo strumento è basato su Stanford CoreNLP e può essere utilizzato come strumento autonomo, includendo le dipendenze in Java all'interno del file "pom.xml".

1.1.2 SimpleNLG

SimpleNLG è una liberia Java che permette di eseguire attività semplici, utili e necessarie per la generazione del linguaggio naturale (NLG). La libreria è stata utile in quanto, grazie ad essa, si sono creati i metodi per la costruzione, seguita poi dalla generazione della frase.
Le classi utilizzate permettono inoltre di specificare le componenti della frase (Soggetto, Verbo, Complemento Oggetto) e anche complementi aggiuntivi (Aggettivi, Avverbi).

Capitolo 2

Implementazione

Una volta definita la vista generale del progetto possiamo passare alla fase implementativa. Si è deciso di utilizzare Java per la soluzione in maniera tale da mantenere una certa continuità con Tint e con SimpleNLG.

2.1 Costruzione struttura dati sintattica

Per costruire l'albero si sono utilizzate le informazioni provenienti dal JSON prodotto da Tint. Infatti, ogni nodo dell'albero presenta queste informazioni; tale scelta è dovuta anche al fatto che, in caso ci siano due occorrenze della stessa parola nella frase, si presenta la possibilità per il sistema di effettuare una discriminazione tra le occorrenze stesse. La struttura è la seguente.

```
public abstract class NodeDependencyTree {
    protected final boolean isRoot;
    protected final Integer indexID;
    protected final String dep;
    protected final String gloss;
    protected String lemma;
    protected String pos;
    protected String glossFather;
    protected NodeDependencyTree father;
    protected Map<Integer, NodeDependencyTree> children;
    protected Map<String, String[]> features;
```

Figura 2.1: Struttura dati utilizzata

Per prima cosa occorre definire i controlli implementati in base alle dipendenze trovate dopo la fase di parsing delle 3 frasi. Le dipendenze sono:

- **aux**: indica la presenza di un verbo ausiliare, modifica le informazioni relative ad un predicato verbale come il tempo/modo;
- **auxpass**: indica la presenza di un verbo di forma passiva, permettendoci di capire se rendere True o False la 'Feature.PASSIVE' del verbo;
- **cop**: indica la presenza di un verbo copulativo. Il nodo figlio, a differenza del caso 'aux' è il verbo da settare, ma come in 'aux', useremo i dati morfologici legati al suddetto nodo per il tempo e la persona.
- **advmod**: indica la presenza di un avverbio, dunque viene modificata la VP, concatenando l'avverbio al verbo mediante il metodo addModifier("avverbio"). In questo modo riusciamo ad ottenere una traduzione più fedele;
- **det/det-poss**: indica un determiner. Se esiste una NP(nounPhrase) viene passato come parametro alla funzione che crea NP. Ad esempio, nlgFactory.createNounPhrase("determiner", "parola");
- **compound**: indica la presenza di un sostantivo composto (spada laser). Anche in questo caso utilizzeremo la funzione addModifier();
- **case**: indica il collegamento con una prepositional phrase;
- **amod**: indica la presenza di un aggettivo modificatore. Si aggiunge come pre modifier "addPreModifier()";
- **nsubjpass**: indica la presenza di soggetto in una frase passiva, quindi viene gestito come fosse l'oggetto della frase.

Inoltre, occorre effettuare anche dei controlli sul nodo attuale:

- **nmod**: indica l'inizio di una NounPhrase. Indica la presenza del complemento di specificazione del complemento oggetto della frase. Nmod, non è quindi da specificare come un post modifier della clausola, ma come un post modifier della NP di cui è figlio nella dipendenza;

- **dobj**: indica l'oggetto di una VP(verb phrase), viene istanziato come una NP.

Successivamente si è passati alla fase di traduzione. Una volta definite le dipendenze incontrate è stato possibile "posizionare" le componenti della frase al proprio posto. Come possiamo vedere dall'immagine che segue, di fondamentale importanza è la classe **TransferTranslationItENG** in quanto al suo interno sono stati implementati i controlli sulle dipendenze e le funzioni di traduzione. Da quella poi viene richiamato il **Realiser** di SimpleNLG per andare a costruire la frase in inglese.

```
System.out.println("Translation");
String translated = TransferTranslationItEng.REALISER.realiseSentence(sentence);
System.out.println("Sentence translated:");
System.out.println(translated);
```

Figura 2.2: Creazione frase tradotta.

Le traduzioni finali sono state:

- è la spada laser di tuo padre - is your father lightsaber;
- ha fatto una mossa leale - made a fair move;
- Gli avanzi della vecchia Repubblica sono stati spazzati via - The last leftover of the old repubblica have been wiped out.

In quest'altra immagine invece viene esplicitato il controllo attraverso cui si va ad aggiungere un aggettivo attraverso il metodo addPreModifier("aggettivo"). Viene presa la parola relativa a quella dipendenza, viene tradotta e poi passata al metodo stesso.

```
if ("amod".equals(posNodeType)) {
    if (translatedChildGloss == null) {
        translatedChildGloss = wordTranslator.apply(child.getGloss());
    } // c&p
    if (nounPhrase == null) {
        prepositionPhrase.addPreModifier(translatedChildGloss);
    } else {
        nounPhrase.addPreModifier(translatedChildGloss);
    }
}
```

Figura 2.3: Controllo sulla dipendenza Amod

Capitolo 3

Conclusioni

Il progetto proposto si è rivelato molto interessante, in quanto ha permesso un approfondimento pratico di quanto appreso durante la prima parte del corso di TLN. Potendo infatti applicare i vari concetti ha portato a capire meglio le fasi di elaborazione di una frase e tutte le parti correlate all'elaborazione stessa.
Il principale obiettivo è stato quello di rendere il programma abbastanza generale, non soffermandosi solo sulla singolarità della traduzione delle tre frasi. A questo proposito si è voluto testare il programma anche su altre frasi di vario tipo: si è notato che esiste una grande quantità di dipendenze tra le parole. Anche da questo si capisce quanto possa essere complessa l'implementazione di un traduttore transfer-sintattico. Infatti il soggetto sottinteso in alcune frasi italiane ci rende difficoltosa la traduzione, in quanto tale regola non esiste nella lingua inglese oppure altri aspetti come l'uso delle forme contratte e l'ordine diverso delle parole in una frase.