Practico 4 - PLN

Implemenentación de un Parser de dependencias Greedy basado en transiciones.

Diego Piloni

Facultad de Matemática, Astronomía, Física y Computación.
Universidad Nacional de Cordoba.

February 25, 2016

Tabla de contenidos

Introducción

Se implementa un Parser de Dependencias, en particular, intentando imitar el modelo voraz propuesto por MaltParser y su algoritmo Nivre arc-eager. `http://www.maltparser.org/`
El algoritmo se encarga de manipular en cada sentencia a parsear dos estructuras distintas, una pila σ, con palabras de la oración con posibles nuevas dependencias y un buffer β, el cual posee las palabras de la oración todavía no analizadas. Con estas estructuras el algoritmo intenta predecir usando un Modelo basado en historias (History-based model for predicting the next parser action) cual de las cuatros posibles transiciones 4 posibles debe ser aplicada.
Estas posibles acciones son:

- SHIFT
- REDUCE
- RIGHT ARC
- LEFT ARC

Introduccion

Siendo las ultimas dos aquellas que generan nuevas dependencias entre palabras de la oración.
Lo que se intenta lograr en este trabajo es hacer una buena traducción entre Arboles de dependencias a secuencias de Historias y acciones, de manera que usando un clasificador, se pueda luego deducir acciones que sean nuevamente traducidas a Arboles de dependencia.
En este trabajo solo se intentará generar dependencias no etiquetadas, quedando pendiente el trabajo de completar el parseo de dependencias etiquetado.

Transiciones

Christopher Manning

Actions ("arc-eager" dependency parser)

Start: $\sigma = [ROOT], \beta = w_1, \ldots, w_n, A = \varnothing$

1. Left-Arc$_r$ $\quad \sigma|w_i, w_j|\beta, A \rightarrow \sigma, w_j|\beta, A \cup \{r(w_j, w_i)\}$

   Precondition: $(w_k, r', w_i) \notin A$, $w_i \neq ROOT$

2. Right-Arc$_r$ $\quad \sigma|w_i, w_j|\beta, A \rightarrow \sigma|w_i|w_j, \beta, A \cup \{r(w_i, w_j)\}$

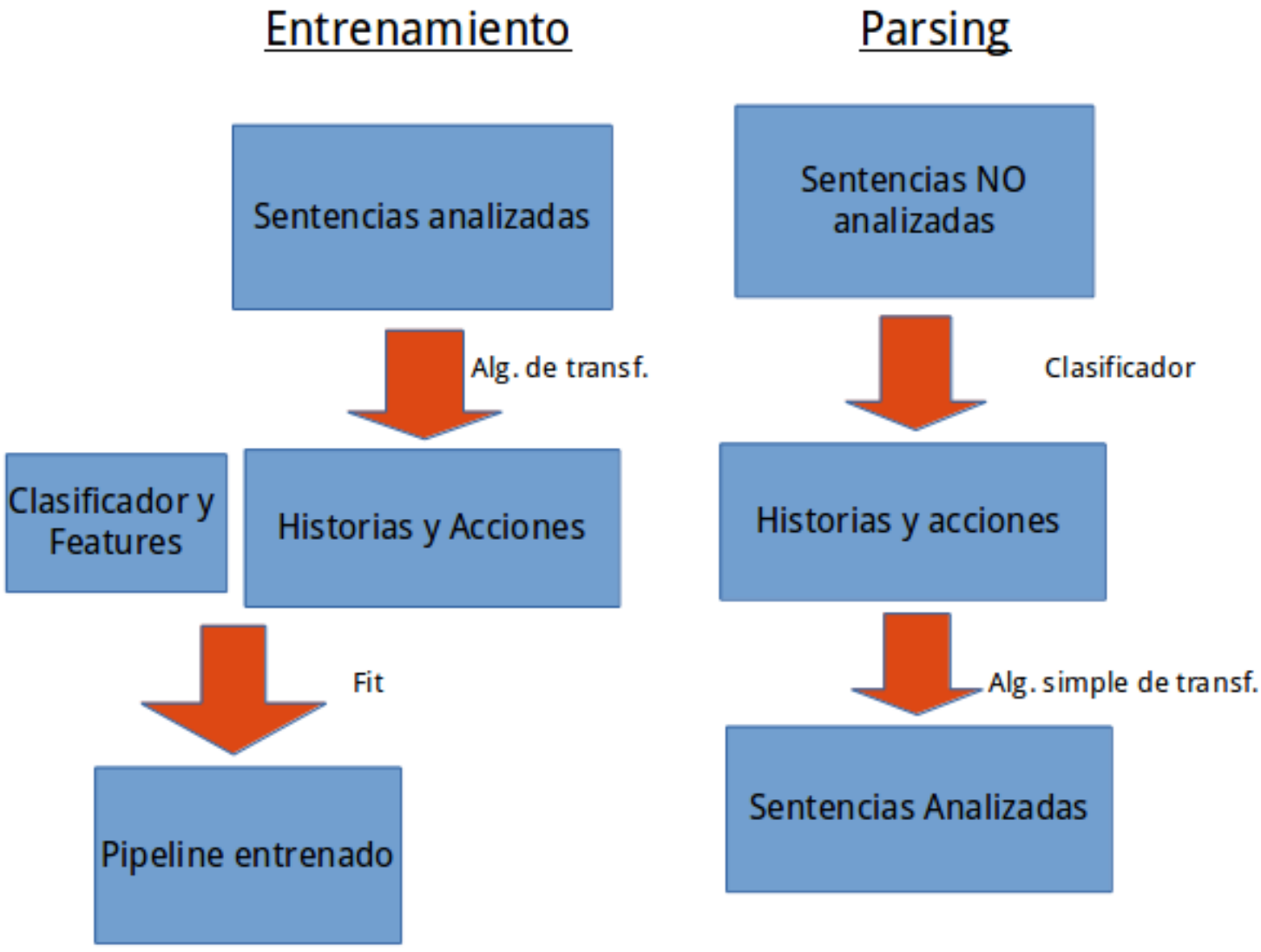
3. Reduce $\quad \sigma|w_i, \beta, A \rightarrow \sigma, \beta, A$

   Precondition: $(w_k, r', w_i) \in A$

4. Shift $\quad \sigma, w_i|\beta, A \rightarrow \sigma|w_i, \beta, A$

Finish: $\beta = \varnothing$

Proceso general



Entrenamiento

Algoritmo para transformar arboles de dependencias en Acciones

$parsed\_sents :: [[INDEX, WORD, \_, POS, HEAD]]$
$\beta \leftarrow parsed\_sents$
$\sigma \leftarrow [ROOT]$ {σ una pila.}
while $\beta \neq \emptyset$ **do**
  if $|\sigma| \geq 1$ **then**
    if Padre de $\sigma[-1]$ es $\beta[0]$ **then**
      LEFT ARC, sacar de σ a $\sigma[-1]$
    else if Padre de $\beta[0]$ es $\sigma[-1]$ **then**
      RIGHT ARC, insertar en σ a $\beta[0]$ y quitarlo de β.
    else
      if Hay arco entre algún elemento de σ y β **then**
        REDUCE, eliminar de σ a $\sigma[-1]$
      else
        SHIFT, Agregar en σ y quitar de β el primer elemento de β
      end if
    end if
  else
    SHIFT, Meter en σ el primer elemento de β y quitar el primero de β.
  end if
end while

Clasificadores y Features

Clasificadores:

- SVM
- LR

Features:

- N Top of stack Words
- N Top of stack Pos
- N Top of buffer Words
- N Top of buffer Pos
- Distance in sent between top of buffer and top of stack
- Pair (top-of-stack word, top-of-buffer-word)
- Pair (top-of-stack-pos, top-of-buffer-pos)

Posibles nuevos features

- Top of stack Leftmost children
- Top of stack Rightmost children
- Top of buffer Leftmost children
- Top of buffer Rightmost children
- ?

Recursos y experimentos realizados

Recursos utilizados: Se utiliza el corpus anotado **Ancora**.
Entrenamiento: (13884 sentencias)

- CESS-CAST-A
- CESS-CAST-AA
- CESS-CAST-P

Evaluación: (3492 sentencias)

- 3LB-CAST

Resultados de experimentación

SVM :

- Global Accuracy: 69.00%
- Tiempo de evaluación: 1 minuto, 5 segundos.

Logistic Regression :

- Global Accuracy: 70.57%
- Tiempo de evaluación: 56 segundos.

Conclusión

- Se pueden apreciar algunas de las ventajas de los metodos data-driven, que es su relativo bajo tiempo de desarrollo tanto como su gran eficiencia en velocidad de parseo con respecto a los metodos dinámicos, como el algoritmo CKY.
- Si bien los resultados obtenidos al experimentar no son tan buenos como se esperan, se cree fuertemente que con una mejor elección de features mejoraría notablemente la Accuracy obtenida.

Bibliografía y Referencias

- Greedy Transition-Based Dependency Parsing - Stanford NLP Video Lectures by Dan Jurafsky, Christopher Manning: https://class.coursera.org/nlp/lecture/177
- http://lrec-conf.org/proceedings/lrec2006/pdf/162_pdf.pdf
- https://spacy.io/blog/parsing-english-in-python
- http://www.maltparser.org/