Natural Language Processing

Estudio realizado por
E. G. Ortiz-García

Índice

1 Introducción

2 Morfología

3 Corrección de errores

4 N-grams

5 Tagging

6 Context-Free Grammar (CFG)

Introducción

- Definición
  - Natural - ¿Para quién?
  - Language - Dice cosas, aunque quizás no las entiendas
  - Processing - Haz todo lo que puedas para darme mucho
- Aplicaciones
  - Extracción de información
  - Motores de búsqueda
  - Traducción
  - Sistemas de reconocimiento de voz
  - Interfaces
  - Clasificación de textos
  - etc.

Introducción

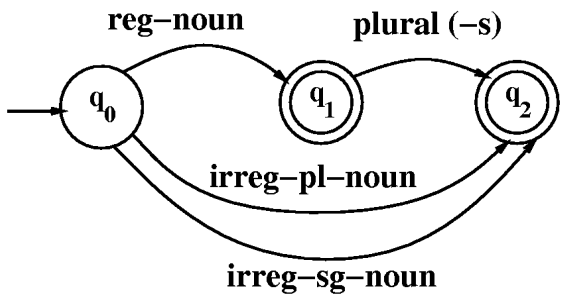
- Competiciones Kaggle
  - EMC Israel Data Science Challenge ($10,000)
  - CPROD1: Consumer products contest 1 ($10,000)
  - Heritage Health Prize ($3,000,000)
  - The Hewlett Foundation: Short Answer Scoring ($100,000)
  - The Hewlett Foundation: Automated Essay Scoring ($100,000)
  - GigaOM WordPress Challenge ($10,000)
  - Detecting Insults in Social Commentary ($10,000)
  - Facebook 2 - Mapping the Internet

Descripción

- Estudio de la formación de las palabras a partir de morfemas.
- Tipos
  - *Inflectional* - Palabras con misma raíz, manteniendo el tipo de palabra resultante.
  - *Derivational* - Fusión de raíz + morfema que forman una palabra de diferente tipo.
- Aplicaciones
  - Parsing morfológico - Descomposición en estructuras simples (raíz, prefijos, sufijos, afijos, tiempo verbal, etc).
    - Reducir tiempo de búsqueda, análisis o agregación de palabras.
    - Reducir almacenamiento.
  - *Stemming* - Búsqueda de las raíces de las palabras.
  - Clasificar POS de cada palabra

Reconocimiento morfológico

- Comprobar si una palabra pertenece al lenguaje o no
- Se necesita
  - Léxico - Conjunto de posibles raíces y afijos
  - *Morphotactics* - Reglas que indican que morfemas puedes seguir a otros
  - Reglas ortográficas o de *spelling* - Modelar cambios en los morfemas cuando estos se unen
- Representación de la morfología
  - Bruta - Cada posible palabra por separado
  - Autómatas de estados finitos



Parsing morfológico

- Transformar una palabra en su raíz más modificadores y vice.
- *Finite State Transducer (FST)*
  - Q - Estados
  - Σ - Conjunto de pares mapeados (i:o)
  - $\delta(q, i : o)$ - Función de transición
- Propiedades
  - Inversión - Intercambio de alfabetos de entra y salida
  - Composición - Se pueden concatenar para formar un FST más complejo
- Transformación entre niveles
  - *Lexical* - Representación mediante morfemas
    Ej. fox + N + PL
  - *Intermediate* Representación con reglas morfológicas incluidas
    Ej. fox + $\wedge$ + s + #
  - *Surface* - Representación con reglas ortográficas incluidas
    Ej. foxes

Descripción

- Objetivo
  - Detectar y/o corregir palabras que probablemente estén mal por otras que probablemente estén mejor.
- Aplicaciones
  - Optical character recognition (OCR)
  - Handwriting recognition
  - Diseño de interfaces
- Tipos
  - Detección de errores en "palabras desconocidas"
  - Corrección de "palabras desconocidas"de manera ciega
  - Detección y corrección basada en contexto

Errores simples

- Basado en un algoritmo en (Kernighan 1990)
- Dos pasos
  - Proposición de correcciones posibles
    - Asume un único error (inserción, borrado, substitución o trasposición) que da lugar a una palabra
  - Scoring de las correcciones
    - Medir $p(t/c)p(c)$
    - c - palabra corregida (corpus palabras)
    - t - Error producido en la palabra (corpus errores)

Errores múltiples

- Minimum Edit Distance
  - Considerar múltiples errores en una palabra
  - Medir distancia entre cadenas
  - Algoritmo basado en programación dinámica
    - Palabra objetivo (t_1^I) y origen (s_1^J)
    - Construir tabla de distancias $d_{i,j} = \text{distance}(t_1^i, s_1^j)$
    - Ecuación de actualización

$$d_{i,j} = \min \begin{cases} d_{i-1,j} + \text{cost inser}(t_i) \\ d_{i-1,j-1} + \text{cost subs}(s_j, t_i) \\ d_{i,j-1} + \text{cost del}(s_j) \end{cases}$$

Descripción

- Objetivo
  - Predecir siguiente palabra
  - Cálculo de probabilidades
- Aplicaciones
  - Reconocimiento de voz, escritura
  - Comunicación aumentativa para discapacitados
  - Detección de errores de *spelling* "severos"
- Métodos
  - Unsmoothed N-grams
  - Smoothed N-grams = Unsmoothed + Smoothing technique

Unsmoothed N-grams

- Basados generalmente en corpus o corpora
  - M - Número de palabras total que lo componen
  - Precisión dependiente del tamaño y diversidad
- Estimaciones
  - $P(w_i) = 1/M$
  - $P(w_i) = c(w_i)/M$
  - $P(w_i/w_{i-N}^{i-1}) = \frac{c(w_{i-N}^{i})}{c(w_{i-N}^{i-1})}$
- ¿Qué pasa si $c(w_{i-N}^{i}) = 0$?

Smoothing techniques

- Add-one
  - $P(w_i) = \frac{c(w_i) + 1}{M + V}$
  - V - Número de palabras distintas (*types*) posibles
  - Problema: Demasiada probabilidad de palabras nunca observadas
- Written-Bell Discounting
  - T - Número de *types* vistas
  - Z - Número de *types* con $c_i = 0$
  - $\sum_{c_i=0} P(w_i) = \frac{T}{T + M}$
  - $P(w_i) = \begin{cases} \frac{T}{Z(M + T)} & c_i = 0 \\ \frac{c_i}{M + T} & c_i > 0 \end{cases}$

Smoothing techniques

- Written-Bell Discounting
  - $T(w_{i-1})$ - Número de *types* vistas que empiezan por w_{i-1}
  - $Z(w_{i-1})$ - Número de *types* con $c_i = 0$ que empiezan por w_{i-1}
  - $$\sum_{c(w_{i-1},w_i)=0} P(w_i/w_{i-1}) = \frac{T(w_{i-1})}{T(w_{i-1}) + M(w_{i-1})}$$
  - $$P(w_i/w_{i-1}) = \begin{cases} \frac{T(w_{i-1})}{Z(w_{i-1})(M(w_{i-1}) + T(w_{i-1}))} & c(w_{i-1}, w_i) = 0 \\ \frac{c(w_{i-1}, w_i)}{M(w_{i-1}) + T(w_{i-1})} & c(w_{i-1}, w_i) > 0 \end{cases}$$

Smoothing techniques

- Good-Turing Discounting
  - N_c - N° de bigrams que aparecen c veces
  - $c^* = (c+1)\frac{N_{c+1}}{N_c}$
  - Cambiando c por c^*

    $P(w_i) = \frac{c^*}{\sum_j c_j^*}$
- Deleted Interpolation
  - $\hat{P}(w_i/w_{i-1}) = \lambda_1(w_{i-1})P(w_i/w_{i-1}, w_i) + \lambda_2(w_{i-1})P(w_i)$

    con

    $\lambda_i >= 0, \sum_i \lambda_i = 1$
  - λ_i estimadas mediante ML con EM en Hold-out corpus

Smoothing techniques

- Katz Back-off
  - Si $c(w_{i-N}^{i}) = 0$, uso $c(w_{i-N+1}^{i})$
  - Utilizando algún método de discounting

$$\hat{P}(w_i/w_{i-1}) = \begin{cases} \dfrac{c^*(w_i, w_{i-1})}{c(w_{i-1})} & c(w_{i-1}, w_i) > 0 \\ \alpha(w_{i-1}) \dfrac{P(w_i)}{\sum_{c(w_i)=0} P(w_i)} & eoc \end{cases}$$

donde

$$\alpha(w_{i-1}) = 1 - \sum_{c(w_{i-1}, w_i)>0} \frac{c^*(w_{i-1}, w_i)}{c(w_{i-1})}$$

Definición

- Objetivo
  - Determinar el POS de cada palabra de una sentencia
- Aplicaciones
  - Paso inicial al análisis sintáctico
  - Sintetizadores de voz
  - Eliminar ambigüedad del significado de palabras
- Tipos de algoritmos
  - Rule-based
  - Stochastic
  - Transformation-based

Stochastic POS Tagger

- Basado en Hidden Markov Models (HMM)
- Dos asunciones:
  - $P(w_i/w_1^{i-1}, t_1^i) = P(w_i/t_i)$
  - $P(t_i/w_1^{i-1}, t_1^{i-1}) = P(t_i/t_1^{i-1})$
- Elegir el tag que maximice
  $p(w_i/t_i)p(t_i/t_1^{i-1})$
- Secuencia de tags más probable
  - Algoritmo de Viterbi
  - Tagged Corpus

Transformation-based Tagger

- Rule-based tagger
  - Diccionario para posibles tags de cada palabra
  - Conjunto de reglas manuales
- Transformation-based tagger
  - Igual a rule-based, pero aprendiendo las reglas (supervisado)
  - Esbozo
    1. Etiquetar con el tag más probable (corpus)
    2. Aprender reglas de cambio (de general a específico)
       Cambiar t_i por t_j si....
    3. Aplicar reglas y comprobar errores
    4. Repetir 2

Palabras desconocidas

- Cómo asignar un tag a una palabra fuera de tu vocabulario
  - Asignar en función de las palabras del entorno
  - Utilizar la distribución de las palabras que ocurren una vez
  - Utilizar información de spelling

Definición

- Representar el modelo generativo de sentencias de un lenguaje.
- Formulación
  - N - Conjunto de símbolos no terminales
    - NP, VP, PP, Nominal, etc
  - Σ - Conjunto de símbolos terminales
    - POS (Nouns, Verbs, Determiners, Adjetives, Adverbs)
  - P - Reglas de producción del lenguaje
    - $A \rightarrow \alpha$ donde $A \in N$ y $\alpha \in N, \Sigma^d$
    - $A \rightarrow w$ donde $A \in \Sigma$ y w una palabra del alfabeto
  - S - Símbolo de inicio

Ejemplo CFG

- Reglas P

  $S \rightarrow$ NP VP

  NP $\rightarrow$ Det Noun

  VP $\rightarrow$ Verb NP

  Det $\rightarrow$ el | un

  Noun $\rightarrow$ conductor | perro

  Verb $\rightarrow$ atropelló

- Sentencias

  el conductor atropelló un perro

  un conductor atropelló un perro

  un conductor atropelló el perro

  el perro atropelló un conductor

Parsing

- Objetivo
  - Analizar una sentencia y asignarle una estructura sintáctica en base a unas reglas CFG
- Aplicaciones
  - Comprobación de errores gramaticales
  - Traducción
  - Respuestas automáticas
- Algoritmos
  - Top-Down Parsing
  - Bottom-Up Parsing
  - Earley Algorithm

Earley Algorithm

- Basado en programación dinámica
- Construcción de N+1 tablas (**c**harts)
  - N - Número de palabras de la sentencia
  - Regla, $[n_1, n_2]$
    - n_1 - Posición de la regla en la sentencia
    - $n_2 - n_1$ - Posición siguiente a analizar en la regla
- Tres procedimientos de construcción
  - Predictor
  - Scanner
  - Completer

Earley Algorithm

- Análisis iterativo sobre los charts y sus elementos
- Predictor
  - Actualiza chart actual
  - Analiza reglas $A \rightarrow \alpha \bullet B\beta, [i, j]$ con $B \notin$ POS
  - Añade todas las reglas $B \rightarrow \gamma$ mediante $B \rightarrow \gamma, [i, j]$
- Scanner
  - Actualiza chart siguiente
  - Analiza reglas $A \rightarrow \alpha \bullet B\beta, [i, j]$ con $B \in \text{POS}[word[j]]$
  - Añade las reglas $B \rightarrow word[j], [j, j + 1]$
- Completer
  - Actualiza chart actual
  - Analiza reglas completas $B \rightarrow \gamma\bullet, [j, k]$
  - Añade todas las reglas en chart j que cumplen $A \rightarrow \alpha \bullet B\beta, [i, k]$

Earley Algorithm

Chart[0]

$\gamma \rightarrow \bullet S$	[0,0]	Dummy start state
$S \rightarrow \bullet NP\ VP$	[0,0]	Predictor
$NP \rightarrow \bullet Det\ NOMINAL$	[0,0]	Predictor
$NP \rightarrow \bullet Proper\text{-}Noun$	[0,0]	Predictor
$S \rightarrow \bullet Aux\ NP\ VP$	[0,0]	Predictor
$S \rightarrow \bullet VP$	[0,0]	Predictor
$VP \rightarrow \bullet Verb$	[0,0]	Predictor
$VP \rightarrow \bullet Verb\ NP$	[0,0]	Predictor

Chart[1]

$Verb \rightarrow book \bullet$	[0,1]	Scanner
$VP \rightarrow Verb \bullet$	[0,1]	Completer
$S \rightarrow VP \bullet$	[0,1]	Completer
$VP \rightarrow Verb \bullet NP$	[0,1]	Completer
$NP \rightarrow \bullet Det\ NOMINAL$	[1,1]	Predictor
$NP \rightarrow \bullet Proper\text{-}Noun$	[1,1]	Predictor

Earley Algorithm

Chart[2]

$Det \rightarrow that\bullet$	[1,2]	Scanner
$NP \rightarrow Det\bullet NOMINAL$	[1,2]	Completer
$NOMINAL \rightarrow \bullet\ Noun$	[2,2]	Predictor
$NOMINAL \rightarrow \bullet\ Noun\ NOMINAL$	[2,2]	Predictor

Chart[3]

$Noun \rightarrow flight\bullet$	[2,3]	Scanner
$NOMINAL \rightarrow Noun\bullet$	[2,3]	Completer
$NOMINAL \rightarrow Noun\bullet\ NOMINAL$	[2,3]	Completer
$NP \rightarrow Det\ NOMINAL\ \bullet$	[1,3]	Completer
$VP \rightarrow Verb\ NP\ \bullet$	[0,3]	Completer
$S \rightarrow\ VP\bullet$	[0,3]	Completer
$NOMINAL \rightarrow \bullet\ Noun$	[3,3]	Predictor
$NOMINAL \rightarrow \bullet\ Noun\ NOMINAL$	[3,3]	Predictor

Y mucho mucho más.

Probabilistic CFGs

Análisis semántico

Machine translating

Global linear models

Bibliografía

- D. Jurafsky, J. H. Martin, *Speech and Language Processing. An introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*, 1999
- M. Collins, *Machine Learning Approaches for Natural Language Processing*, http://www.ai.mit.edu/courses/6.891-nlp/, 2003