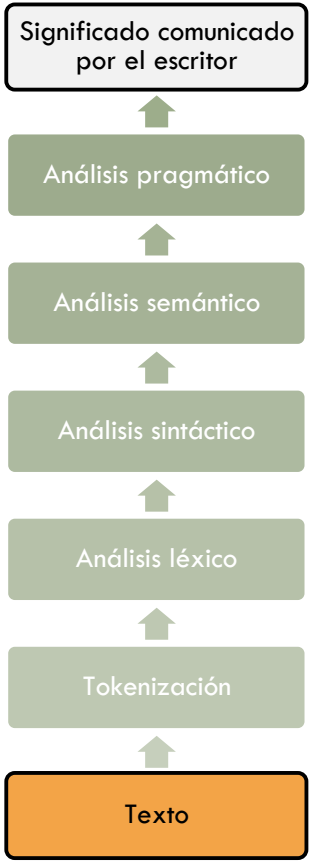
Análisis de textos
Análisis de caracteres

Álvaro Barbero Jiménez

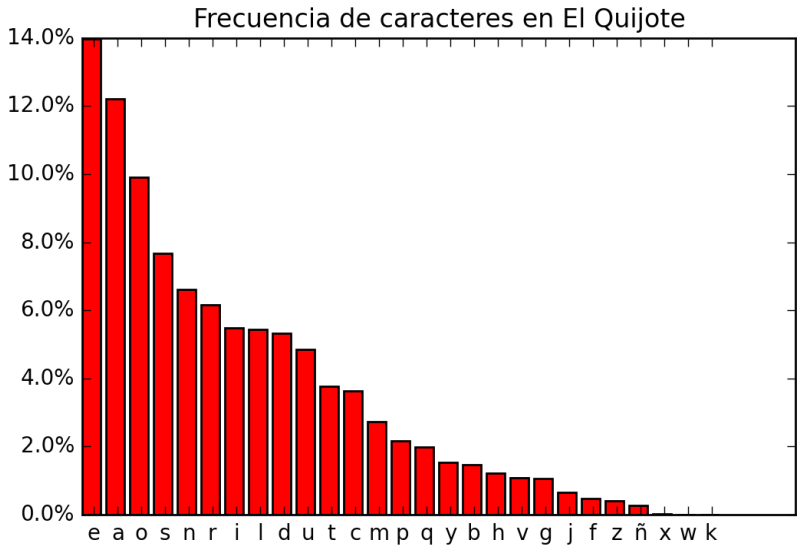
Punto de partida: texto digital

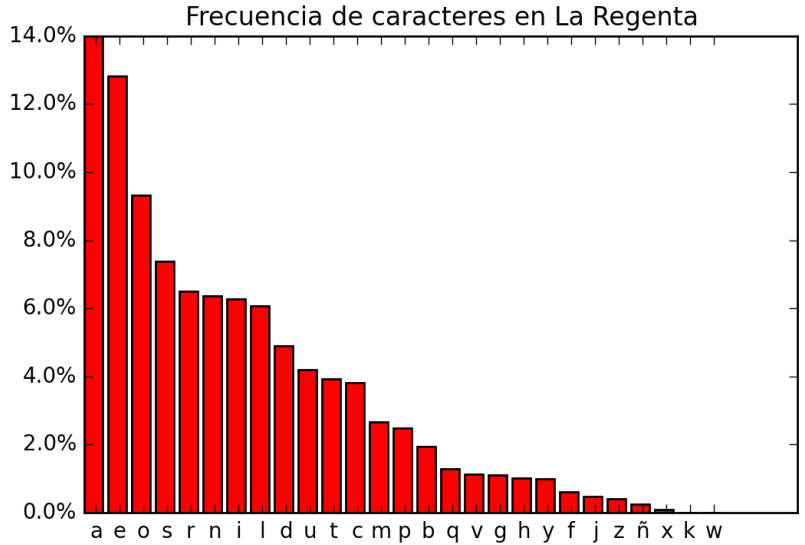
- Asumimos que para todos los procesos de análisis subsiguientes el texto ya se encuentra en un formato digital apropiado
  - Sigue una codificación reconocida: ISO-8859, UTF-8, …
  - Libre de marcadores de formato: etiquetas HTML, XML, …
  - Libre de otros elementos ajenos al texto: imágenes, hipervínculos, …
- De no ser así, existen herramientas para la conversión, filtrado y tratamiento de textos
- Teniendo únicamente el texto y sin análisis ya pueden extraerse algunas características básicas



Frecuencia de caracteres

- Resumir un texto como la frecuencia con la que aparece cada carácter





- Muy simple, pero efectivo en algunos casos
- Útil en idiomas de tokenización difícil (Chino, Japonés, …)

Frecuencia de grupos de caracteres

- Considerar grupos de 2, 3, … n caracteres consecutivos y contar sus frecuencias
  - También conocido como n-gramas de caracteres
  - Robusto frente a errores de escritura, formatos no convencionales (ej. mensajes móviles)

El veloz murciélago hindú comía feliz cardillo y kiwi

el	2	o_	2	lo	2
l_	1	El	1	wi	1

- Ojo: coste de almacenamiento exponencial en n
  - Para un idioma con C caracteres, el coste es $O(C^n)$

Aplicación: identificación de idioma

- Objetivo: dado un texto, identificar en qué idioma está escrito
  - Muy útil para luego poder hacer los análisis lingüísticos adecuados al idioma
- Características útiles para el problema
  - Caracteres: frecuencia, n-gramas, skip-gramas
- Clasificadores empleados
  - Naive Bayes
- Recursos
  - Implementación con un 99% de acierto en 53 idiomas: https://github.com/shuyo/language-detection

Identificación de idioma: ejemplo

sprogregistrering

???

språkgjenkjenning

???

الكشف عن اللغة ???

تشخیص زبان ???

زبان کی شناخت ???

Shuyo, Nakatani. Language Detection Library for Java. http://code.google.com/p/language-detection/

Complejidad de Kolmogorov

- La complejidad de información de un objeto cualquiera (texto, imagen, etc…) puede medirse mediante la complejidad de Kolmogorov
- La complejidad de Kolmogorov K(A) de un objeto A es el tamaño del programa mínimo que puede producir ese objeto
  - En general, un programa en una máquina de Turing o lenguaje descriptivo universal, en particular, un programa en cualquier lenguaje de programación
- Ejemplos
  - A = "abababababababababababababababab"
    - for i=1:10, print("ab") → K(A) = 23
  - A = "4c1j5b2p0cv4w1x8rx2y39umgw5q85s7"
    - print("4c1j5b2p0cv4w1x8rx2y39umgw5q85s7") → K(A) = 41

Complejidad de Kolmogorov condicionada

- Dados dos objetos A y B, la complejidad de Kolmogorov condicionada K(B | A) es el tamaño del programa más pequeño posible que recibiendo A como entrada puede generar B
- Si A es similar a B el programa puede hacer uso de ello para generar B con más fácilmente
- Ejemplos
  - A = "abababababababab"
  - B = "abababababababababababababababab"
    - print(A, A) → K(B | A) = 11
  - A = "abababababababab"
  - B = "4c1j5b2p0cv4w1x8rx2y39umgw5q85s7"
    - print("4c1j5b2p0cv4w1x8rx2y39umgw5q85s7") → K(B | A) = 41 = K(A)
- Propiedad de simetría de la información

$$K(A,B) = K(A) + K(B|A) = K(B) + K(A|B)$$

Distancia de Kolmogorov

- A través de la complejidad de Kolmogorov puede definirse una distancia entre objetos cualquiera, que compara cómo de diferentes son
- Distancia de información normalizada

$$e(A,B) = \frac{\max\{K(A|B), K(B|A)\}}{\max\{K(A), K(B)\}}$$

- Alternativamente y usando simetría de información

$$e(A,B) = \frac{\max\{K(A,B) - K(B), K(A,B) - K(A)\}}{\max\{K(A), K(B)\}} = \frac{K(A,B) - \min\{K(B), K(A)\}}{\max\{K(A), K(B)\}}$$

- Problema: la complejidad de Kolmogorov K no es computable

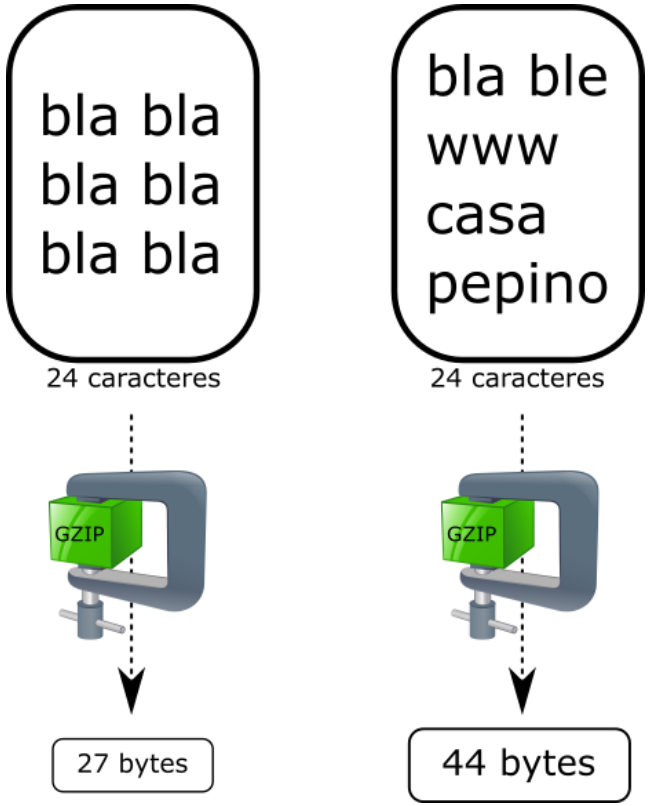
Distancia de compresión normalizada

- Encontrar la complejidad de Kolmogorov de un objeto A implica encontrar el programa más pequeño posible capaz de reproducir A
  - Esto es equivalente a encontrar la mejor compresión posible sin pérdida del objeto A
  - Esto es, la complejidad de Kolmogorov de un objeto nos da el mejor algoritmo de compresión que puede existir para ese objeto
- Como aproximación a ese algoritmo de compresión óptimo podemos usar métodos de compresión generales ya conocidos: ZIP, RAR, GZIP, …
  - Z(A) = tamaño comprimido del objeto A
- Distancia de compresión normalizada

$$e_Z(A,B) = \frac{Z(A,B) - \min\{Z(B), Z(A)\}}{\max\{Z(A), Z(B)\}}$$

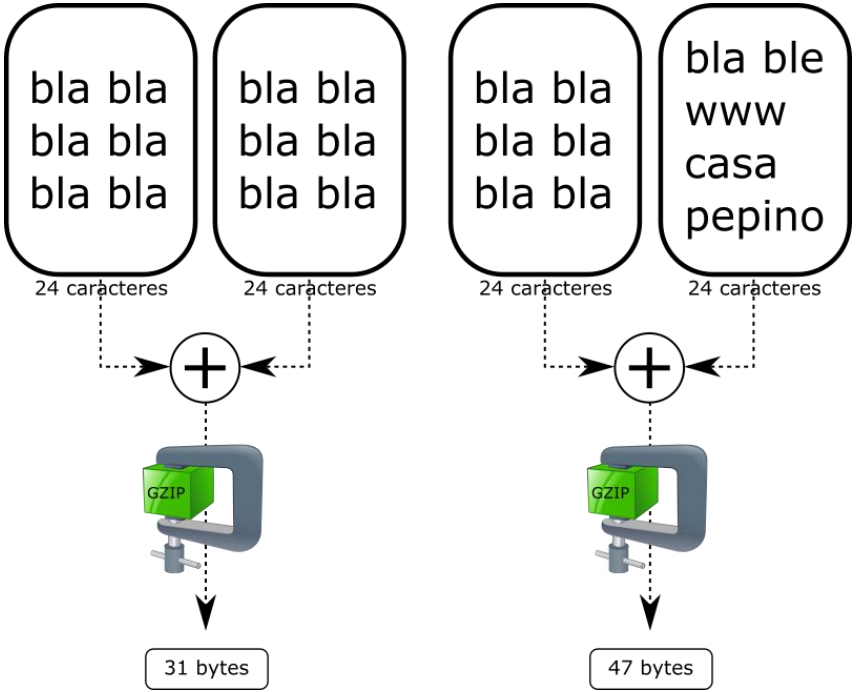
Complejidad de compresión y riqueza del texto

- Los métodos de compresión (ZIP, RAR, etc…) reducen el tamaño de un fichero en base a explotar información redundante, y reexpresarla de una forma más eficiente
- Un texto con mucho vocabulario e información redundante (baja complejidad de Kolmogorov) se podrá comprimir más que un texto con mayor diversidad
- El ratio de compresión es un indicador de la riqueza del texto
  - Poco tamaño tras compresión: poca riqueza
  - Mucho tamaño tras compresión: alta riqueza
- RAR funciona muy bien para esta aplicación (no comprime en bloques)



Distancia de compresión: comparativa de textos

- Considerando un texto o documento como un objeto, podemos usar la distancia de compresión normalizada para estimar la similitud entre textos
  - Al concatenar dos textos similares se obtiene un nuevo texto con mucha información redundante, facilitando la compresión
  - Si los dos textos son muy distintos, se minimiza la cantidad de información redundante



Álvaro Barbero Jiménez

@albarjip

https://github.com/albarji

albarji.deviantart.com