PYTHON PARA LINGÜISTAS

INTRODUCCIÓN A NLTK:

PROCESAMIENTO DE CORPUS BÁSICO. TOKENIZACIÓN.

ALEJANDRO ARIZA

CENTRE DE LLENGUATGE I COMPUTACIÓ

UNIVERSITAT DE BARCELONA

LINGÜÍSTICA Y PYTHON

- Lingüísta con Python (hasta ahora):
  - Los strings representan datos de texto
  - Las listas pueden almacenar diferentes unidades de texto: párrafos, frase, palabras...
  - Un corpus puede ser cargado directamente de un fichero de texto
- El resto del curso:
  - Recursos disponibles
  - Funciones especializadas en procesamiento de texto
  - Aplicaciones: etiquetado, análisis sintáctico, clasificación de texto
- Estas nuevas herramientas se construyen sobre lo que conocemos hasta la fecha

NLTK: THE NATURAL LANGUAGE TOOLKIT

- Un paquete de Python externo:
  - Aplicación de terceros
  - No incluida cuando instalas Python
  - Después de instalarla, puedes importarla al igual que hacemos con csv, collections, etc

    ```
    import nltk
    ```
- Podéis leer más acerca de NLTK en su web http://www.nltk.org/

NLTK (2)

- Por qué usar NLTK:
  - Un paquete muy completo y desarrollado
  - Muy didáctico → NLTK Book http://www.nltk.org/book/
  - Dirigido tanto a programadores como lingüistas, estudiantes, graduados o investigadores
- Algunas debilidades:
  - Más lento que otras herramientas más modernas
  - No pensado para entornos de producción
- Otras herramientas más modernas y óptimas para NLP – Spacy

COMPONENTES NLTK
DESCARGA DE RECURSOS

- NLTK está instalado ya en vuestros ordenadores así que podéis importarlo directamente

```
import nltk
```

- Hay recursos adicionales disponibles con NLTK (corpus, diccionarios, modelos pre-entrenados)
- Los recursos adicionales requieren ser descargados usando la función download() de NLTK

```
nltk.download()
```

ACCESO A CORPUS IN NLTK

- NLTK incluye una variedad de corpus tanto procesados como sin procesar
- Para acceder a un corpus, debemos importarlos desde el paquete corpus al igual que hacemos con las librerías:

  ```
  from nltk.corpus import reuters
  ```

- Cada corpus suele ser un conjunto de múltiples ficheros (e.g.: artículos de noticias)

  ```
  reuters.fileids()
  ```

- Podéis procesar los ficheros uno por uno (usando el id de fichero) o todos juntos.. Si no introducís ningún id de fichero, NLTK os devolverá el corpus completo.

CORPUS PRE-PROCESADO

- NLTK contiene corpus preprocesados y sin preprocesar
- Algunos corpus en NLTK ya están tokenizados, etiquetados con PoS y/o analizados sintácticamente
- El corpus sin preprocesar es la data del “mundo real” (un string de caracteres)

  `reuters.raw(id)`
- La versión tokenizada de un corpus (words()) ya tiene identificados los tokens individuales

  `reuters.words(id)`

ACCESO A HERRAMIENTOS DE NLTK

- NLTK tiene muchas otras herramientas útiles para procesar un corpus
- La mayoría se pueden acceder a través de funciones
- Una vez importamos nltk, podemos acceder a sus funciones internas usando nltk.FUNC_NAME()

  nltk.word_tokenizer()

- Recuerda: el operador punto (.) te permite acceder a lo que contiene ese objeto

PROCESAMIENTO BÁSICO DE CORPUS

- En esta clase, aprenderéis los pasos necesarios para un procesamiento básico de corpus
- Al menos los pasos que queremos aplicar a la mayoría de los corpus
- Hasta la fecha ya hemos realizado algunas de estas tareas manualmente
- Veremos cómo se utilizan las herramientas de NLTK para conseguir estos pasos

PROCESAMIENTO BÁSICO DE CORPUS: LIMPIEZA

- El primer paso en el procesado de un corpus de texto es la limpieza
- Muy a menudo el corpus proviene de Internet. Puede incluir HTML o XML caracteres no legibles, imágenes, links, hashtags.
- He limpiado de Corpus puede consistir en:
  - Borrar información innecesaria (dependerá de la tarea)
  - Convertir el corpus a la codificación correcta (utf-8) y borrar data no legible.
- El Corpus que vamos a utilizar esta ya limpiado por lo que no necesitamos hacerlo de nuevo

PROCESAMIENTO BÁSICO DE CORPUS: TOKENIZACIÓN. SEGMENTACIÓN DE FRASES.

- Un string es una lista de caracteres, que no contiene ninguna unidad explícita lingüística
- Normalmente, el procesamiento de lenguaje, queremos trabajar con unidades lingüísticas en vez de caracteres: palabras, expresiones multi palabras, frases, etc.
- Tokenización: convertir un string de caracteres en una lista de tokens
- Segmentación de frases: identificar el comienzo y final de una frase. Convertir una lista de Tokens en una lista de listas de Tokens

TOKENIZACIÓN.
NAÏVE SPLITTING. EXPRESIONES MULTI-PALABRA.

- Nosotros ya experimentamos con la herramienta básica de tu organización - .split()
- El método .split() puede ser usado con cualquier String para convertirlo a una lista
- El comportamiento por defecto de .split() es separar el texto por espacios.
- Sin embargo, la tokenización (incluso en inglés) no es tan simple:
  - “New York” – “New guy”, “12:54” – “he said: hello!”,
  - “09/03/2020” – “not recommended for cats/dogs/rabbits”, “O’Neil” – “John’s”
- NLTK contiene una función word_tokenize() que se encarga de este problema

ESTADÍSTICA BÁSICA DEL CORPUS

- Lo más simple que podemos hacer con un corpus es contar:
  - Contar el número de tokens en el corpus
  - Contar el número de tokens únicos en el corpus
  - Contar la frecuencia tokens únicos
- Esta información puede ser útil para muchas tareas
- Nosotros ya hemos contado la frecuencia de tokens únicos usando un simple bucle
- NLTK nos proporciona una mejor función para esta tarea: FreqDist()

N-GRAMAS

- N-Gramas constituyen una noción muy popular en NLP y CL
- N-Gramas Son secuencias de Tokens con una longitud definida. Por ejemplo, un bi-grama es una secuencia de dos Tokens, un Tri-grama es una secuencia de tres Tokens
- Por ejemplo: “Esto es una frase corta” contiene los siguientes bigramas:

  (“Esto”, “es”), (“es”, “una”), (“una”, “frase”), (“frase”, “corta”)

- Podemos obtener listas de n-gramas usando las funciones de NLTK bigrams() trigrams() ngrams()

MODELOS DE LENGUAJE

- “Juan va al dentista cuando le duelen las...”
- Hipótesis: Si tenemos suficiente contexto, podemos “predecir” lo que alguien va a decir a continuación.
- La comedia, ironía o sarcasmo a menudo “juegan” con estas expectativas.
- Un modelo de lenguaje estadístico predice: 1) la probabilidad de una palabra dado el contexto; o 2) la probabilidad de todo el texto
- Los modelos de lenguaje son muy populares en NLP

LA SUPOSICIÓN DE MARKOV. N-GRAMAS.

- El tamaño del contexto es un problema de los modelos de lenguaje
- La suposición de Markov – no es necesario todo el contexto, solo necesitamos una parte:: “Nueva” -> “York”, “aléjate” -> “de” ...
- El concepto de n-gramas viene de la suposición de Markov
- La combinación de palabra y estadísticas de n-gramas nos da una distribución de probabilidad

PROCESAMIENTO BÁSICO Y LINGÜÍSTICA

- Un procesamiento básico de corpus (frecuencia, n-gramas, co-ocurrencia) consiste principalmente en aplicar herramientas que provienen de la estadística y programación básica para extraer información
- No hace uso de mucha información lingüística (excepto tokenización)
- Para ciertas tareas y lenguajes, un procesado básico es suficiente
- Para muchas otras tareas y lenguajes, no es suficiente
- Algunas de estas tareas (e.g.: tokenización) pueden ser mucho más complicadas en lenguajes diferentes al inglés

TOKENIZACIÓN AVANZADA. EXPRESIONES REGULARES.

- Una forma muy común de realizar tokenización es usar expresiones regulares
- Las expresiones regulares son un lenguaje para encontrar patrones específicos:

  [a-zA-Z]+ – cualquier secuencia de caracteres que contenga únicamente letras entre a-z y A-Z

  [0-9]* – una secuencia de 0 o más dígitos

  e-?mail – tanto “e-mail” como “email”

  ([A-Z]\.)+ – Abreviaciones, e.g. U.S.A.

GRAMÁTICA DE EXPRESIONES REGULARES

- NLTK proporciona un tokenizador con expresiones regulares RegexpTokenizer()
- Un ejemplo simple de expresión regular puede ser:

```
(?:[A-Z]\.)+            # abreviaciones, e.g. U.S.A.
| \w+(?:-\w+)*          # palabras con guiones opcionales
| \$?\d+(?:\.\d+)?%?    # cantidades y porcentajes, e.g. $12.40, 82%
| \.\.\.                # elípsis (...)
| [][.,;"'?():-_`]      # algunos caracteres especiales
```

¡GRACIAS!
¿PREGUNTAS?