Prácticas de SAR

Sistemas de Almacenamiento y Recuperación de información

Práctica 3: El Mono Infinito

El mono infinito

Descripción del problema

Figura: monkey at work

El teorema del mono infinito afirma que un mono pulsando teclas ***al azar*** *sobre un teclado durante* ***un periodo de tiempo infinito*** *casi seguramente podrá escribir finalmente cualquier libro que se halle en la Biblioteca Nacional de Francia.*

Objetivo de la práctica

Ya que no tenemos tanto tiempo, crearemos un programa en python que procese un documento y que utilice la información extraída de él para ayudar al mono a escribir su libro.

Ejercicio

¿Qué debo hacer?

Se proporcionan tres ficheros:

- **SAR_p3_monkey_indexer.py**: crea un índice a partir de un fichero de texto.
- **SAR_p3_monkey_info.py**: crea un fichero con información a partir de un índice.
- **SAR_p3_monkey_evolved.py**: genera frases **al estilo** de un índice.

y una plantilla que se debe completar:

- **SAR_p3_monkey_lib_plantilla.py**

¿Qué debo hacer?

Debes completar **Y DOCUMENTAR** los métodos *index_sentence*, *compute_index* y *generate_sentences* de la clase **Monkey**, incluida en la librería *SAR_p3_monkey_lib.py*, para conseguir las funcionalidades de los 3 programas que se detallan a continuación.

¿Qué debe hacer? (SAR_p3_monkey_indexer.py)

Ejecución

python SAR_p3_monkey_indexer.py spam.txt

Funcionalidad

- SAR_p3_monkey_indexer.py:
  1) Recibe como argumento el nombre de un fichero de texto,
  2) lo divide en frases, las tokeniza y crea un índice donde acumula estadísticas de qué palabras siguen a otras, y
  3) Guarda el índice en un fichero binario con el mismo nombre del fichero de texto pero con extensión ".index".

¿Qué debe hacer? (SAR_p3_monkey_indexer.py)

Tokenización:

- La separación entre frases vendrá dada por “.”, “;”, “!”, “?” o por dos saltos de línea.
- Se eliminarán todos los símbolos no alfanuméricos.
- Los tokens serán las palabras del documento en minúsculas.
- Se añadirá un símbolo especial “$” que indique inicio y final de frase.

Creador de índices:

- El índice se guardará como una tabla hash (diccionario de python).
- Las claves del diccionario serán los tokens del documento más la palabra especial “$” .
- Cada entrada del diccionario contendrá:
  - El número total de veces que ha aparecido el token.
  - Una lista con todas los tokens que han aparecido en el documento después de la clave (incluido “$”) y la frecuencia.
- La lista de sucesores debe estar ordenada por el número de apariciones del par de tokens.

¿Qué debe hacer? (SAR_p3_monkey_indexer.py)

Ejecución

python SAR_p3_monkey_indexer.py spam.txt

“spam.txt”:

Egg and Bacon;
Egg, sausage and Bacon;
Egg and Spam;
Spam Egg Sausage and Spam;
Egg, Bacon and Spam;
Egg, Bacon, sausage and Spam;
Spam, Bacon, sausage and Spam;
Spam, Egg, Spam, Spam, Bacon and Spam;
Spam, Spam, Spam, Egg and Spam;
Spam, Spam, Spam, Spam, Spam, Spam, Spam, baked beans, Spam, Spam, Spam and Spam;
Lobster Thermidor aux crevettes with a Mornay sauce, garnished with truffle pate, brandy and a fried egg on top and Spam

¿Qué debe hacer? (SAR_p3_monkey_info.py)

Ejecución

python SAR_p3_monkey_info.py spam.index

Funcionalidad

- SAR_p3_monkey_info.py:
  1. Recibe como argumento el nombre de un fichero de índice,
  2. crea un fichero de texto con la extensión ".info" con información del índice.

¿Qué debe hacer? (SAR_p3_monkey_info.py)

python SAR_p3_monkey_info.py spam.index

“spam.info”:

```
####################
#       INFO       #
####################
filename: 'spam.txt'

####################
#     BIGRAMS      #
####################
$   =>  11  =>  spam:5 egg:5 lobster:1
a   =>  2   =>  mornay:1 fried:1
and =>  12  =>  spam:9 bacon:2 a:1
aux =>  1   =>  crevettes:1
bacon   =>  6   =>  sausage:2 and:2 $:2
.
.
.
spam    =>  27  =>  spam:11 $:9 egg:3 bacon:2 baked:1 and:1
thermidor   =>  1   =>  aux:1
top =>  1   =>  and:1
truffle =>  1   =>  pate:1
with    =>  2   =>  truffle:1 a:1
```

¿Qué debe hacer? (SAR_p3_monkey_evolved.py)

Ejecución

python SAR_p3_monkey_evolved.py spam.index

Funcionalidad

- SAR_p3_monkey_evolved.py:
  1. Recibe como argumento el nombre de un fichero de índice y opcionalmente un número entero como segundo argumento,
  2. Utiliza la información del índice para generar frases **al estilo** del índice,
  3. Genera 10 frases si no se le indica una cantidad como segundo argumento.

¿Qué debe hacer? (SAR_p3_monkey_evolved.py)

Ejemplo de ejecución

```
> python SAR_p3_monkey_evolved.py spam.index 7

egg and a fried egg on top and bacon
egg spam spam spam egg spam bacon sausage and spam spam spam baked beans spam
lobster thermidor aux crevettes with truffle pate brandy and spam spam
egg bacon and a mornay sauce garnished with truffle pate brandy and spam
spam spam bacon
spam
egg on top and bacon sausage and spam spam
```

¿Qué debe hacer? (SAR_p3_monkey_evolved.py)

¿Cómo se inicia cada frase?

- Se elige como palabra inicial '$'.

¿Cómo se elige cada palabra siguiente?

- Las palabras siguientes se eligen sucesivamente de forma "**aleatoria ponderada**" entre las sucesoras de la palabra actual teniendo en cuenta el número de veces que ha aparecido.

¿Cuándo se termina una frase?

- La palabra siguiente elegida es la palabra "final" especial ($), o
- se llega a un número máximo de palabras, 25 en nuestro caso.

Cosas útiles

Guardar objetos python en un fichero

pickle

```
import pickle

def save_object(obj, filename):
    with open(file_name, 'wb') as fh:
        pickle.dump(obj, fh)

def load_object(filename):
    with open(file_name, 'rb') as fh:
        obj = pickle.load(fh)
    return obj
```

Números “aleatorios” en python

librería random

```
import random


random.randint(a, b)

    "Return a random integer N such that a <= N <= b."


random.choice(seq)

    "Return a random element from the non-empty sequence seq. If seq is empty,
    raises IndexError."
```

Ampliación

python SAR_p3_monkey_indexer.py spam.txt tri

Añadir un segundo argumento 'tri' para añadir al índice grupos de trigramas.

* Se creará otro índice en el que cada entradas será una tupla ($w1$, $w2$) que enlazará con las palabras vistas detrás de $w1$ y $w2$.

* Si se utilizan trigramas el nombre **del** índice debe terminar en "_tri.index".

SAR_p3_monkey_info.py spam_tri.index

Se deberá modificar para mostrar también la información del índice de trigramas.

SAR_p3_monkey_evolved.py spam_tri.index

La generación de frases se realizará teniendo en cuenta el índice de trigramas.

* ¿Cuál es la primera palabra de cada frase?
* ¿Y la segunda?