PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN
IIC2233 - PROGRAMACIÓN AVANZADA

Actividad 15

2^o semestre 2017
16 de noviembre de 2017

Web Services & Expresiones Regulares

Introducción

Como en época de exámenes tu tiempo libre tiende asintóticamente a cero , decidiste automatizar el proceso de encontrar pareja, creando Prinder, un Tinder automático. Para lograr esto, utilizarás la API de reconocimiento facial de Microsoft.

Lo que deben realizar

A través de la consola, el usuario deberá ser capaz de elegir entre distintos atributos que quiere en su pareja. Estos pueden ser: género, color de pelo, uso de lentes, etcétera. Luego, tu aplicación Prinder, deberá ser capaz de retornar como *output* la dirección de correo y la URL de la imagen de la(s) persona(s) que cumplan con los requisitos.

Para desarrollar la aplicación, se te entregará un archivo llamado `faces.txt` donde se encuentran descripciones personales, un correo UC y URLs de imágenes de las personas que pertenecen a la aplicación. Este archivo viene *sucio*, por lo que en un principio deben limpiarlo siguiendo lo indicado en la **Parte 1**.

Luego, deberás obtener las características de las imágenes utilizando la API de *Microsoft Azure* para cada persona presente en el archivo. Para poder obtener información, **debes** tener la *subscription key*. En la **Parte 2**, se explica con más detalle lo que deberás hacer.

Finalmente, deberás realizar la selección de personas según los atributos indicados por el usuario. Esto se explica en la **Parte 3**.

Para esta actividad, un ejemplo del *output* puede ser lo siguiente:

Parte 1: Limpiar el archivo

Cada línea del archivo `faces.txt` contiene un par `<url>;<descripción>` que representa a una persona de la aplicación. Las descripciones tienen caracteres que no corresponden a letras y que deberán ser eliminados. Para ello, deberás utilizar expresiones regulares.

Para limpiar el archivo, deberás implementar una *regex* para desarrollar los siguientes casos:

- Cada vez que un carácter `$` está acompañado de uno o más números, debes cambiar todo esto por un único espacio. Por ejemplo, `$3141592` se cambia por un espacio; `$0` también se cambia por un espacio.
- Cada vez que un carácter `$` está acompañado de **dos** letras mayúsculas a la derecha, debes extraer completamente este trío de caracteres. Por ejemplo, `$PC` se reemplaza por un *string* vacío.

Una vez que lo hayas limpiado, deberás guardar el resultado en el archivo `faces_clean.txt`, manteniendo el mismo formato que `faces.txt`.

Parte 2: Obtener las características de las imágenes

Para obtener las características de las personas, deberás utilizar la API de *Microsoft Azure* de reconocimiento facial.

Uso de la API

Antes de comenzar a hacer *requests*, generalmente las APIs requieren realizar un par de trámites previos y averiguar detalles para realizar la solicitud con el formato correcto. Lo que hace esta API es leer la foto contenida en una URL y retorna las características **físicas** de la persona en la imagen.

- Página para obtener el *subscription key*: `https://azure.microsoft.com/es-es/try/cognitive-services/#vision`
- Documentación: `https://goo.gl/rTEV8X`

Para esta actividad, deberás hacer una *request* a la API a través del método HTTP, `POST`. Para realizar la solicitud, simplemente hay que enviar la URL de las imágenes que están en el archivo `faces.txt`. Para que el resultado sea exitoso, deberás utilizar los siguientes parámetros:

1. URL a la que se le hace la request:
   `https://westCentralus.api.cognitive.microsoft.com/face/v1.0/detect?returnFaceAttributes=<ATRIBUTOS>`

2. `ATRIBUTOS`: `gender, hair, smile, glasses, <atributo_libre>`. La API logra identificar muchos más rasgos de los solicitados. Busca en la documentación uno de estos para usarlo en tu programa.
3. *Header*: {`‘‘Content-Type’’: ‘‘application/json’’,‘Ocp-Apim-Subscription-Key’’: «TOKEN»` }
4. *Data*: { `‘‘url’’: LINK_FOTO` }

Nota: El ***header*** debe ser un diccionario, y la ***data*** debe ser un *string*. La información que retorne la API debe ser procesada para guardar solo lo necesario para las consultas de su programa. En el siguiente enlace, puedes ver un ejemplo de la información que retorna la API:

- Página de la API: `https://azure.microsoft.com/es-es/services/cognitive-services/face/`

Parte 3: seleccionar personas

Tu programa debe tener un menú sencillo en la consola. En este, el usuario debe ingresar los atributos que está buscando en una persona y tu programa debe buscar, las que tengan las características. El programa debe mostrar en consola las características físicas y la descripción de las personas que haya encontrado.

El usuario solo podrá elegir entre los siguientes rasgos:

- `str` – Género
- `str` – Color de pelo
- `bool` – Si usa o no lentes
- `bool` – Sonrisa ($\geq$ 0,5)
- `?` – Atributo libre que hayas escogido en la sección anterior.

Nota: Para cada atributo, la API ofrece una cantidad de valores de distintos tipos. Por ejemplo, la sonrisa puede ser un número real entre 0 y 1, dependiendo de cuán sonriente aparece la persona. Sin embargo, para efectos de producir el *match*, se tomará si es sonriente (igual o superior a 0,5) o no lo es (inferior a 0,5) —y por eso lo tomaremos como un `bool`. Por otra parte, “si usa o no lentes” también lo transformaremos a un `bool`; no obstante, la API ofrece una serie limitada de *strings* que ustedes deben mapear al valor booleano.

Requerimientos

Uso de API:

- (1.60 pts) Correcta realización de solicitudes a *Face API*
  - (0.80 pts) Se utilizan los parámetros correctos para llamar a la API.
  - (0.80 pts) Se lee el resultado de la API y se guarda lo necesario para las consultas.
- (2.60 pts) Realizar *match*:
  - (2.40 pts) Asignar los atributos correctamentes por cada foto:
    - (0.40 pts) Género
    - (0.40 pts) Color de pelo
    - (0.40 pts) Uso de lentes
    - (0.40 pts) Sonrisa
    - (0.80 pts) Atributo libre
  - (0.20 pts) Retornar el enlace y descripción de la foto con la persona elegida, que calce con lo solicitado.

Regex:

- (0.60 pts) Primera limpieza con expresiones regulares: `$` con números.
- (0.60 pts) Segunda limpieza con expresiones regulares: `$` con letras.

CLI – *command-line interface*:

- (0.60 pts) Interfaz en consola para interactuar con el programa.

Entrega

- **Lugar:** En su repositorio de GitHub en la **carpeta** `Actividades/AC15/`
- **Hora:** 16:55
- Si está trabajando en pareja, basta con que un miembro suba la actividad. Si se suben actividades distintas, se corregirá una de las dos al azar.