Respuestas

1. **¿Qué requisitos debe cumplir el archivo dataset-original.ttl para que su programa funcione adecuadamente?**

   Para que el enriquecedor funcione correctamente, el archivo *dataset-original.ttl* debe tener declaradas las URI de los individuals que están definidas como sujeto en el *owl:sameAs* del archivo *links.ttl* . En caso contrario, el programa no fallará pero puede omitir información.

2. **¿Cree que funcionará si le damos como entrada el archivo generado por alguno de sus compañeros?**

   Sí, en caso de que se cambie el *dataset-original.ttl* , para que el programa funcione se deben ejecutar las siguientes acciones:

   1. Se deberá generar un nuevo archivo de *links.ttl* , mediante el comando: `python3 main.py --links`
   2. Ejecutar el enriquecedor, mediante el comando: `python3 main.py -e ../data/dataset-original.ttl ../data/links.ttl`

   Por otro lado, en caso de que se cambien los dos archivos (*dataset-original.ttl* y *links.ttl*) consideramos que no habría problema. Ya que por la forma en la que desarrollamos el algoritmo no necesitamos conocer el vocabulario subyacente utilizado. Además no modificamos los nombres desginados (normalizados) para los actores por lo que tampoco tendríamos problema durante la inserción de los datos.

   Logramos intercambiar varios datasets con algunos compañeros, no hubo ningún problema. Los datasets, junto con su correspondientes archivos enriquecidos se encuentran en la carpeta `data/{Nombre de compañero/}`

3. **¿Cómo efectuó la búsqueda de correspondencias entre su dataset y dbpedia?**

   Por cada tripleta `<twss_actor> rdf:type dbo:Actor` de nuestro dataset, se matchea la tripleta `<twss_actor> owl:sameAs <dbr_actor>` de *links.ttl* . Luego con rdflib se carga el grafo de `dbr_actor` , se extraen las tripletas correspondientes, se cambia el sujeto por `<twss_actor>` y se agrega la información al dataset enriquecido.

4. **¿Se podría automatizar la búsqueda de correspondencias? ¿Cómo? ¿Con qué efectividad?**

   Sí, se podría automatizar la búqueda de correspondencias. Para ello simplemente hay que recorrer todos los objetos del predicado `owl:sameAs` e ir recolectando la información como se describió anteriormente. En caso de no tener el predicado `owl:sameAs` definido se nos ocurre hacer un tratamiento sintáctico y convertir nuestra URI en la URI requerida, tal como lo hicimos para generar el archivo *links.ttl* . Aunque esto puede llegar a tener errores, la efectividad va a depender de la ontología con la que estamos tratando, sobre todo si tiene patrones en su URI. Por ejemplo, dbpedia define sus URI con nombres separados con guión bajo en camel case, lo que hace que la conversión resulte sencilla.

5. **Le pedimos que incluya la información obtenida de dbpedia en el archivo resultante. Desde el punto de vista de alguien que va a utilizar su dataset, ¿era necesario incluir esa información o alcanzaba con solo proveer los links sameAs?**

   Depende del caso que estemos tratando, si queremos "congelar" la información entonces tiene sentido que hagamos lo que hicimos, en caso contrario, no tiene mucho sentido que copiemos la información de dbpedia ya que referenciando el objeto del owl:sameAs se puede obtener la misma información y mucho más. También puede servir si queremos solo representar un cierto grado de información y por alguna razón se contradice con la demás información que contiene dbpedia, pero hay que tener en cuenta se debe ir actualizando. Por ejemplo pensemos en las ocupaciones del actor, si el actor comienza a trabajar en nueva película y fue cargada en dbpedia, en nuestro caso no la veríamos reflejada, ya que "congelamos" la información al copiarla directamente desde dbpedia, en cambio si hubieramos usado el owl:sameAs esa información se podría conseguir simplemente referenciando el objeto de owl:sameAs.