Revista Tecnología y Ciencia
ISSN 1666-6933 DOI: https://doi.org/10.33414/rtyc.33.45-64.2018

Universidad Tecnológica Nacional
Septiembre-Diciembre 2018 / Año 16 - Nº 33

Búsqueda de Respuestas como Aplicación del Problema de Extracción de Relaciones

Alejandra Carolina Cardoso
Facultad de Ingeniería e IESIING, Universidad Católica de Salta, Salta
acardoso@ucasal.edu.ar

M. Alicia Pérez Abelleira
Facultad de Ingeniería e IESIING, Universidad Católica de Salta, Salta
aperez@ucasal.edu.ar

Enzo Notario
Facultad de Ingeniería e IESIING, Universidad Católica de Salta, Salta
enzo.notario@gmail.com

Recepción 18/11/2016
Aprobación 20/07/2017

Resumen

El volumen de información no estructurada en forma de documentos de texto de diversos orígenes es cada vez mayor. Para poder acceder a ella y obtener conocimiento que pueda ser aplicado a una diversidad de tareas, primero hay que "estructurar" esa información. La extracción de la estructura relacional entre entidades, en forma de tripletas basadas en un verbo, puede ser aplicada al problema de búsqueda de respuestas. Este trabajo ha hecho uso de técnicas eficientes de análisis superficial de texto y el sistema construido tiene una precisión y recall comparable a otros sistemas del estado del arte. Las tripletas extraídas forman la base de conocimientos sobre la que se hacen consultas para obtener respuestas a preguntas en lenguaje natural. Los resultados de este sistema de búsqueda de respuestas sobre un banco de preguntas a un corpus de 2052 documentos sobre Salta obtenidos de la web demuestran la validez de este enfoque.

Palabras claves: extracción de relaciones, búsquedas de respuestas, minería de textos

Abstract

The volume of unstructured information in the form of text documents from various sources is ever increasing. In order to access it and obtain knowledge that

Revista Tecnología y Ciencia
ISSN 1666-6933

can be applied to a variety of tasks, you must first "structure" that information. The extraction of the relational structure between entities, in the form of triplets based on a verb, can be applied to the problem of finding answers. This work has made use of efficient techniques of shallow text analysis. The system built has a precision and recall comparable to other state of the art systems. The extracted triplets form the knowledge base on which queries are made to obtain answers to questions in natural language. The results over a question database to posed to a corpus of 2052 web documents on Salta demonstrate the validity of this approach.

Keywords: relation extraction, question answering systems, text mining

Introducción

Un sistema de búsqueda de respuestas tiene como objetivo encontrar respuestas en lenguaje natural a preguntas también en lenguaje natural. Para ello dispone de grandes cantidades de texto que son analizadas bien en el momento de responder a la pregunta, o bien procesando el corpus de texto de antemano y construyendo una base de conocimientos. El problema de búsqueda de respuestas, y el campo más amplio de la minería de textos, han ido aumentando en importancia e interés con la disponibilidad cada vez mayor de grandes bases de documentos, incluida la web. En una primera etapa exploramos la búsqueda de respuestas a preguntas factoides sencillas con un corpus de más de 8000 documentos correspondientes a resoluciones rectorales de una universidad (Cardoso, Bini, & Pérez Abelleira, 2015). Este corpus había sido anotado previamente con un sistema de reconocimiento de entidades con nombre generado mediante técnicas de aprendizaje automático. Las preguntas del usuario se clasificaban según el tipo de respuesta esperada y se convertían en consultas a un motor de búsqueda semántica. Las respuestas obtenidas son entidades del tipo buscado que responden a la pregunta, mostradas en el contexto de texto en que aparecen. El presente trabajo también explora el problema de la búsqueda de respuestas pero como una aplicación del problema de la extracción automática de relaciones de grandes cantidades de texto. Así el problema de búsqueda de respuestas se convierte en la búsqueda en una base de conocimientos de relaciones previamente extraídas y en la extracción de la respuesta a la pregunta a partir de las relaciones relevantes. A diferencia del trabajo anterior, el corpus de documentos es obtenido de la web y las preguntas pueden ser más complejas.

Las secciones 2 y 3 describen la tarea de extracción de relaciones y las técnicas y algoritmos desarrollados en este trabajo para dicha tarea. En la Sección 4 se aplica la búsqueda de respuestas, seguido de su evaluación experimental en la Sección 5 y finalizando con algunas conclusiones.

Extracción de relaciones

En los últimos años ha habido mucho interés en la posibilidad de extraer proposiciones básicas de grandes volúmenes de texto. Esto es una instancia de la tarea de extracción de relaciones dentro de la minería de texto que consiste en reconocer la

aserción de una relación semántica entre dos o más entidades, típicamente procedentes de texto y el desafío es poder reconocer relaciones que no han sido predefinidas de antemano (Banko & Etzioni, 2008).

Frecuentemente las relaciones se almacenan en forma de tripleta. Esta forma captura una representación semántica superficial que sirve de representación intermedia a la hora de estructurar el contenido del texto y como tal puede ser punto de partida para otras tareas de la minería de texto y la comprensión del mismo, tales como textual entailment (implicación textual), llenar huecos en el texto, población de ontologías o generación de otras bases de conocimiento. En nuestro caso las tripletas son la representación básica del texto para obtener respuestas a preguntas en lenguaje natural en los sistemas de búsquedas de respuestas (Gamallo, 2014).

Uno de los enfoques más usados para este problema es el denominado OpenIE (Open Information Extraction). El resultado de la extracción son tripletas basadas en un verbo. Cada tripleta <arg1; rel; arg2> corresponde a una proposición y consta de un verbo rel y dos argumentos y pretende capturar una relación importante en una oración. Una oración puede dar lugar a más de una tripleta. La extracción de relaciones no está limitada a un conjunto pre-especificado de tipos de relaciones, sino que los tipos y estructura de las relaciones se descubren a partir del texto, por ejemplo para la oración "Respecto a la ciudad, el cerro San Bernardo tiene una altura de 284.92 m según indica el mojón ubicado en la cima del mismo" un sistema de OpenIE podría extraer la tripleta

<el cerro San Bernardo; tiene; una altura de 284.92 m>

que es coherente, mientras que la tripleta

<el cerro San Bernardo; tiene; una altura de 284.92 m según indica el mojón ubicado en la cima del mismo>

también es coherente pero seguramente estaría sobre-especificada para ser útil en tareas posteriores como implicación semántica, o población de una ontología. Otras tripletas posibles no deberían ser extraídas por no ser coherentes, tales como

<respecto a la ciudad; tiene; el cerro San Bernardo>

<el cerro San Bernardo; tiene; el mojón ubicado en la cima del mismo>

<el mojón, ubicado; en la cima del mismo>

<una altura de 284.92 m; indica; el mojón>

<respecto a la ciudad; indica; el mojón>

Revista Tecnología y Ciencia
ISSN 1666-6933

Nótese que un análisis sintáctico profundo y análisis semántico nunca obtendría estas últimas tripletas. Sin embargo, los sistemas de OpenIE evitan el costo del análisis profundo y se basan en características superficiales de las palabras, por lo que es más probable que surjan relaciones incoherentes como las anteriores. Las relaciones incoherentes son generalmente eliminadas mediante el uso de heurísticas

Gamallo (Gamallo, 2014) presenta una organización de los sistemas de OpenIE a lo largo de dos dimensiones. La primera tiene que ver con los patrones que se utilizan para detectar relaciones: unos sistemas usan datos de entrenamiento para aprender automáticamente un clasificador que es capaz de extraer relaciones, otros se basan en patrones generados "a mano". En la línea de investigación de OpenIE se han utilizado tres enfoques para extraer relaciones:

Aprendizaje auto-supervisado: consiste en etiquetar relaciones usando heurísticas y distant supervision, aprender a partir de esos ejemplos de relaciones un extractor de relaciones, y extraer las relaciones, detectan-do en el texto pares de elementos candidatos a ser argumentos de una relación y aplicando el extractor para detectar la relación y construir la tripleta (Etzioni, Banko, & Cafarella, 2006) (Wu & Weld, 2010). Estos sistemas detectan demasiadas relaciones que son incoherentes, debido a que el extractor primero detecta los argumentos y luego busca la relación entre ellos(Fader, Soderland, & Etzioni, 2011).

Análisis del contexto: se extraen relaciones no solo centradas en un verbo, sino otros tipos de relaciones binarias, e incluso con más de dos argumentos (Mausam, Schmitz, Bart, Soderland, & Etzioni, 2012). Se utiliza un análisis sintáctico más profundo que en el caso anterior que requiere más tiempo y recursos computacionales para cantidades grandes de texto.

Uso de restricciones sintácticas y léxicas en forma de reglas predefinidas: es un enfoque intermedio entre los anteriores. Las relaciones están centradas en el verbo, su extracción comienza con la detección del verbo o frase verbal, y después se buscan sus posibles argumentos. El análisis realizado está basado en el etiquetado de partes del habla (POS) que es menos exigente computacionalmente que el análisis más profundo del caso anterior. Ejemplos son los sistemas ReVerb (Fader, Soderland, & Etzioni, 2011) y ExtrHech (Zhila & Gelbukh, 2013).

La segunda dimensión mencionada por Gamallo se refiere al tipo de técnicas de análisis sintáctico utilizadas: unos sistemas utilizan técnicas de análisis sintáctico superficial (etiquetado POS, chunking) y otros técnicas más sofisticadas que realizan un análisis sintáctico más profundo, tales como dependency parsing.

El sistema de extracción de relaciones descripto en el presente trabajo se encuadra en el tercer enfoque de la primera dimensión, ya que utiliza patrones elaborados a mano. En cuanto a la segunda dimensión, se realiza un análisis sintáctico superficial.

Arquitectura del sistema

El proceso típico de una aplicación de búsqueda incluye tres fases: (a) extracción del contenido, (b) su correspondiente indexación en una base de conocimientos, y (c) búsqueda en el índice de la base de conocimientos. En el enfoque elegido en este trabajo, basado en la extracción de relaciones, y con el objetivo último de dar

respuestas a preguntas textuales del usuario, estas fases genéricas se especifican de la siguiente forma:

a) construcción de la base de conocimientos mediante el procesamiento de una gran cantidad de documentos. En este proyecto los documentos son archivos de texto extraídos de la web.

b) extracción de las relaciones del texto en forma de tripletas e indexación usando un motor de búsqueda. En este trabajo se usó Lucene.

c) búsqueda: dada una pregunta planteada por un usuario en lenguaje natural la búsqueda de respuestas se realiza convirtiendo primero la pregunta en una consulta al índice construido en la fase anterior. El resultado de la búsqueda es un conjunto de tripletas. Cada tripleta tiene asociados la oración y el documento de los que procede. La respuesta(s) a la pregunta se extraen de la tripleta, y la oración y el documento al que pertenecen se usan como contexto para presentar la respuesta al usuario.

Para la implementación y experimentos se utilizó la arquitectura UIMA (Unstructured Information Management Architecture) (Ferrucci & Lally, 2004). Se trata de una arquitectura basada en componentes para construir sistemas de procesamiento de información no estructurada. En UIMA, el componente que contiene la lógica del análisis se llama anotador. Cada anotador realiza una tarea específica de extracción de información de un documento y genera como resultado anotaciones, que son añadidas a una estructura de datos denominada CAS (common analysis structure). El pipeline de UIMA encadena los anotadores en secuencia para realizar una tarea que incluye la extracción de texto, el procesamiento del mismo, y el almacenamiento apropiado de los resultados. La Figura 1 muestra el proceso de extracción de relaciones de un corpus de documentos de texto, correspondiente a las fases (a) y (b) mencionadas anteriormente, que han sido implementadas como un pipeline de UIMA, una secuencia de anotadores que se escriben en las secciones que siguen.

Figura 1. Proceso de extracción de relaciones del corpus de documentos

Adquisición del texto

Este anotador procesa la colección de archivos extraídos de la web. Se trata de archivos de texto en que se encuentran, delimitados por secuencias de caracteres especiales, la URL de la página, el nombre del archivo local de texto, el título de la página, y los párrafos de texto. En estos archivos se eliminan las etiquetas HTML, etc. Es común en textos de la web que estén divididos en secciones. Dado

Revista Tecnología y Ciencia
ISSN 1666-6933

que el encabezado de la sección puede contener información importante sobre el contexto del contenido de la sección, se guarda el texto dividido en secciones con sus encabezados correspondientes, si los hubiera. La extracción de los diversos elementos se realiza mediante expresiones regulares. El resultado se guarda como una "anotación", la estructura de datos básica de UIMA, y se entrega al anotador siguiente.

Separación en oraciones

Cada sección del texto es dividida en oraciones. El producto de este anotador es un conjunto de oraciones, almacenadas como anotaciones UIMA que incluyen también el título de la página y el título de la sección y metadatos como el nombre del archivo de origen y la URL del documento.

Anotador de etiquetas POS

Como se indicó anteriormente el enfoque de nuestro sistema para la extracción de relaciones utiliza un análisis sintáctico superficial, basado en etiquetas POS (part of speech). Para generar esas etiquetas se utilizó el etiquetador POS de FreeLing una colección de herramientas de código abierto para el análisis del lenguaje .

Etiqueta	Descripción	Ejemplo
AQ	Adjetivo	alegre, grande
RG	Adverbio	ahora, siempre
RN	Adverbio negativo	no
DA	Artículo	la, los
NC	Nombre común	casa,
NP	Nombre propio	Argentina, Pedro
VM	Verbo	canta, somos
VA	Verbo auxiliar	ha, hemos
VG	Verbo gerundio	cantando
VP	Verbo participio	cantada, cantados
VN	Verbo infinitivo	cantar,
VS	Verbo semiauxiliar	soy, es
PO	Pronombre	me, se, lo
PX	Pronombre posesivo	mío, nuestro
PP	Pronombre personal	yo, usted
PD	Pronombre demostrativo	aquel, esa
PI	Pronombre indefinido	algo, alguien
SP	Preposición	al, con, a partir de
Z	Número	239, veinte, 74 %

Tabla 1. Etiquetas EAGLES utilizadas

El anotador POS toma cada oración anotada y, realiza el análisis morfológico con FreeLing obteniendo para cada palabra su lema y etiqueta EAGLES , que identifica cada palabra como artículo, verbo, adjetivo, etc. Cada etiqueta EAGLES está formada por seis o siete caracteres que representan tipo de palabra, género, número, etc. En este trabajo son suficientes los primeros caracteres de la etiqueta (categoría y tipo) para identificar cada palabra. La Tabla 1 muestra las etiquetas utilizadas, un subconjunto de las etiquetas generadas por Freeling, y la Tabla 2 (segunda columna) muestra las etiquetas obtenidas en el análisis de una oración de ejemplo.

Los verbos en participio son un caso particular interesante. Considérese por ejemplo la oración La ciudad, conocida por su clima fresco, tiene una magnífica vegetación. Freeling etiqueta la palabra conocida como verbo participio pero en el contexto de la extracción de relaciones es más conveniente considerar además la función del participio como de adjetivo, no solo como verbo. Por ello en este caso el anotador genera dos posibles etiquetas POS para esta palabra: adjetivo y verbo, y ambos casos son considerados para la extracción de tripletas.

A partir de la secuencia de etiquetas POS la oración se divide en fragmentos más grandes, o chunks, centrados en un verbo (VP-chunk) o en un nombre (NP-chunk). El comienzo de un chunk se marca con B y cada elemento restante del mismo con I. La etiqueta O se usa para las palabras restantes de la oración, que no forman parte de un chunk, tales como pronombres interrogativos o signos de puntuación. La tercera columna de la Tabla 2 muestra los chunks en la oración de ejemplo.

Por último se generan las estructuras de datos que se pasarán al siguiente anotador, el extractor de tripletas. Se trata de tres vectores que capturan las etiquetas POS y lemas generados por FreeLing y la división en chunks. Nótese que en el caso de los verbos se almacena el lema (verbo en infinitivo). Más adelante, a la hora de generar las consultas, se buscarán sinónimos de los verbos mediante un diccionario. Para ello es más eficiente manejar el verbo en infinitivo.

Extracción de relaciones en forma de tripletas

La	**DA**	**NP**
provincia	**NC**	
de	**SP**	
Salta	**NP**	
tiene	**VM**	**VP**
una	**DI**	**NP**
población	**NC**	
de	**SP**	
535.303	**Z**	
habitantes	**NC**	
.	**Fp**	**O**

Tabla 2. Etiquetas POS y chunks para una oración de ejemplo

Como punto de partida para la extracción de relaciones se toman los tres vectores generados por el anotador anterior: los tokens o palabras de la oración, sus etiquetas POS, y la división en chunks (verbales y nominales).

El anotador comienza extrayendo la relación, o frase verbal, centrada en torno al verbo. La extracción está basada en la restricción sintáctica de ReVerb(Fader, Soderland, & Etzioni, 2011) a la que se han aplicado diversas modificaciones. Reverb obtiene una frase verbal según el patrón V | VP | VW*P (un verbo, un verbo seguido de una preposición, o un verbo seguido de nombres, adjetivos o adverbios y finaliza con preposición). Nuestro sistema extiende el patrón de frase verbal. La Figura 2 muestra el patrón utilizado formado con las etiquetas EAGLES.

ReVerb toma como frase verbal la secuencia más larga de palabras que satisface el patrón. Nuestro sistema considera además todas las secuencias de palabras posibles que satisfacen el patrón.

$V \mid VW^*P+$
V = RN? P0? VA? (VM| VS |VP) (VN| VP)?
W = (NC| NP| AQ | AO| RG | PD| PX| PI |DA|Z)
P = SP (VN | VG)?
donde:
* representa repetición cero o más veces
+ representa repetición una o más veces
? representa repetición cero o una vez

Figura 2. Patrones para detectar la frase verbal en una oración

Por ejemplo dada la oración El cerebro realiza millones de actividades el texto más largo equiparado como frase verbal por el patrón de ReVerb es realiza millones de (VW*P). Nuestro anotador selecciona todas las ocurrencias en el texto del patrón, en este caso realiza (V) y también realiza millones de (VW*P).

Los elementos RN? PO? VA? se agregan para considerar frases que no necesariamente empiecen con un verbo. Por ejemplo en la oración La Ruta Nacional 40 no se encuentra pavimentada siendo apta sólo para vehículos todoterreno se obtiene la frase verbal no se encuentra pavimentada, donde no se corresponde con la etiqueta RN? y se con la etiqueta PO?, algo que aparentemente ReVerb no permite.

La frase verbal obtenida r se almacena en la tripleta con el verbo en infinitivo, como ya se indicó, y eliminando los pronombres (se, le, etc.).

Una vez encontrada la frase verbal de una oración, se pasa a localizar sus argumentos. Para cada frase verbal r obtenida se buscan chunks a la izquierda y a la derecha de la frase verbal (x e y respectivamente). Un chunk es la secuencia de palabras más larga que no contiene un verbo y por lo menos tiene un nombre común o propio. De esta forma se obtiene la primera tripleta candidata <x; r; y>, que es la tripleta con la mayor cantidad de palabras. Esta tripleta inicial se agrega al conjunto de tripletas T obtenidas de la oración siendo analizada.

Véase por ejemplo la oración De la médula espinal nacen los nervios periféricos, que permiten movimientos voluntarios e involuntarios, sensaciones y reflejos. Para la frase verbal permiten el chunk a la izquierda que es la secuencia más larga sin un verbo y por lo menos con un nombre es los nervios periféricos, que se convierte en el primer argumento x. El chunk a la derecha es movimientos voluntarios e involuntarios, sensaciones y reflejos, segundo argumento y. La tripleta resultante es por lo tanto:

<los nervios periféricos; permiten; movimientos voluntarios e involuntarios,

sensaciones y reflejos>. Nótese que en esta oración existe otro verbo, nacen, que da lugar a su vez a otra tripleta.

La tripleta inicial es procesada de distintas maneras para obtener tripletas adicionales como sigue. Un argumento arg se divide para obtener nuevas tripletas según las siguientes heurísticas:

h1: si arg contiene texto entre paréntesis, este texto se elimina, obteniendo arg1.

h2: si arg contiene una conjunción (y, o, e) o contiene uno o más separadores (; o :) se divide arg en los fragmentos de texto a ambos lados del separador(es) arg2, arg3, ...

Este proceso se realiza para ambos argumentos x e y. Se generan tripletas para r y cada una de las combinaciones de fragmentos x1, x2 , x3 ... con y1, y2 , y3 ...
En el ejemplo anterior las tripletas resultantes son

<los nervios periféricos; permiten; movimientos voluntarios e involuntarios>

<los nervios periféricos; permiten; sensaciones>

<los nervios periféricos; permiten; reflejos>

Es posible construir tripletas unarias, es decir, con un solo argumento. Es de destacar que Reverb no considera dichas tripletas unarias, pero hemos encontrado que son de utilidad cuando la extracción de tripletas se integra en la búsqueda de respuestas.

Cada nueva tripleta obtenida se agrega a T si no se ha agregado ya (no se almacenan tripletas repetidas) y si cada argumento contiene por lo menos un nombre común o nombre propio. A modo de ejemplo, tomando la oración La República Argentina (conocida simplemente como Argentina) es un país de América del Sur, ubicado en el extremo sur y sudeste de dicho subcontinente, se obtienen las tripletas de la Tabla 3.

Nótese que la heurística que considera el participio como verbo mencionada en el anotador anterior lleva a tripletas erróneas, como son las últimas de la tabla en que el participio dicho se convierte en una relación.

Las heurísticas descritas generan un gran número de tripletas, algunas de ellas redundantes, incoherentes o hasta incorrectas, como se ve en el ejemplo. Sin embargo, dada la aplicación de las tripletas a responder preguntas, como se describirá a continuación, se prefiere un gran número de tripletas para aumentar el recall aún a riesgo de disminuir la precisión.

Indexador de tripletas

Revista Tecnología y Ciencia
ISSN 1666-6933

A medida que se van obteniendo las tripletas para cada una de las oraciones del texto, éstas se van indexando para poder ser utilizadas por un motor de búsqueda. Para ello se ha utilizado la librería Lucene de indexación y búsqueda. La unidad de texto indexada por Lucene es la tripleta, es decir, un "documento" de Lucene se corresponde con una tripleta. La tripleta ha sido obtenida por el anotador anterior y almacenada como una anotación de UIMA con los siguientes campos: relación, arg1, arg2, oración de la que extrajo, título del documento en el que aparece la oración, título de la sección en el documento si éste está dividido en secciones, nombre de archivo y URL de procedencia del texto. Cada uno de estos campos es indexado en Lucene. Los cinco primeros campos son indexados como campos de texto, en los que se puede buscar, y los restantes solo como metadatos.

Antes de indexar un campo de texto, éste es preparado mediante un analizador. En particular eliminación de caracteres no deseados (como marcas de HTML), descomposición del texto en palabras (tokenización), conversión a minúscula, eliminación de palabras superfluas (o stopwords como el, la, que, y, ...). Se ha utilizado el Standard Analyzer de Lucene para este proceso.

x	*r*	*y*	
La República Argentina	ser	un país de América del Sur , ubicado en el extremo sur y sudeste de dicho subcontinente	primera tripleta. Verbo en infinitivo
La República Argentina	ser	un país de América del Sur	(h2) se divide arg2 por coma y por y
La República Argentina	ser	ubicado en el extremo sur	
La República Argentina	ser	sudeste de dicho subcontinente	
un país de América del Sur	ubicar en	el extremo sur	*Ubicado* es considerado por Freeling verbo participio También se aplica h2
un país de América del Sur	ubicar en	el extremo sur y sudeste	
un país de América del Sur	ubicar	en el extremo sur	
un país de América del Sur	ubicar en	sudeste	
en el extremo sur	decir	subcontinente	*Dicho* es considerado por Freeling como verbo conjugado
en el extremo sur y sudeste	decir	subcontinente	
sudeste	decir	subcontinente	

Tabla 3. Tripletas obtenidas a partir de la oración de ejemplo

Búsqueda de respuestas

una vez que se dispone de la base de conocimientos, es decir, de la colección de relaciones adecuadamente indexada, comienza la fase de búsqueda de respuestas propiamente dicha.

Dada una pregunta planteada por un usuario en lenguaje natural, debe ser convertida primero en una consulta adecuada para el motor de búsqueda construido sobre la API de Lucene(Carlson, 2013). El proceso tiene similitudes con el descripto para la extracción de tripletas. Primero se realiza el etiquetado POS de la pregunta y se ubican

los marcadores de la misma (pronombres interrogativos, ya que se trata de preguntas factuales). A continuación se detectan el verbo y los chunks nominales que lo preceden o siguen. Pueden haber ninguno, uno o más de un chunk a cada lado del verbo.

Con esos elementos se aplican una serie de heurísticas para construir una consulta al índice obtenido de las tripletas. La primera heurística consiste en buscar los elementos de la pregunta (verbo, chunks antes y después del verbo) respectivamente en los elementos (verbo, arg1, arg2) de la tripleta. Las siguientes heurísticas van reduciendo las restricciones en la consulta, especialmente si no se han encontrado respuestas a las versiones anteriores de la consulta. Un ejemplo es ampliar la consulta no solo al verbo que aparece en la pregunta, sino también a sus sinónimos. A continuación se enumeran dichas heurísticas.

i. buscar tripletas con verbo v, combinaciones de los chunks que aparecen antes del verbo en el arg1 de la tripleta y de los que aparecen después en el arg2 de la tripleta
ii. buscar otras combinaciones de los lugares de aparición (arg1 y arg2) de los diversos chunks
iii. lematizar las palabras que aparecen en plural en los chunks para permitir una equiparación aproximada. Por ejemplo, "asentamientos" se convierte en "asentamiento~2", lo cual indica que se busca una aproximación del lema asentamiento
iv. incorporar sinónimos del verbo a los casos anteriores
v. eliminar el verbo de la consulta
vi. eliminar los adjetivos de la consulta eliminar alguno de los argumentos de la consulta
vii. ampliar la búsqueda de los elementos de la pregunta no solo a la tripleta sino también al título del documento en que aparece la tripleta

A continuación se muestran ejemplos de la aplicación de cada heurística. Para cada ejemplo se muestra:

(a) la pregunta en lenguaje natural,
(b) el resultado del procesamiento de la misma usando las etiquetas POS,
(c) la consulta en el lenguaje de consultas de Lucene que produce la tripleta de respuesta. Para simplificar se ha colocado en cada ejemplo solamente la consulta en el lenguaje de Lucene más sencilla que obtiene una respuesta. La consulta presentada al índice es una conjunción de una serie de consultas obtenidas aplicando las heurísticas.
(d) la primera tripleta devuelta por el motor de búsqueda,
(e) la respuesta mostrada al usuario. Los metadatos de la tripleta se utilizan para presentar al usuario el contexto en que aparece la respuesta: la oración, el título de la página y su URL. La respuesta se muestra resaltada en el

Revista Tecnología y Ciencia
ISSN 1666-6933

contexto de la oración en que aparece.

Un buscador Lucene devuelve un ranking para cada resultado. En el caso en que haya varias oraciones entre las tripletas devueltas, este ranking se usa como heurística para ordenar las respuestas y mostrar al usuario la que más se acerque a lo buscado.

El primer ejemplo, Figura 3, muestra la aplicación de la primera heurística. Se trata del caso más sencillo. La palabra ley que aparece antes del verbo se busca en el campo arg1 de la tripleta (término +arg1:ley en el lenguaje de consultas de Lucene), las palabras del chunk que aparece después del verbo se buscan en el campo arg2 (término +arg2:bandera +arg2:salta), y el verbo en el campo rel (término +rel: crear).

El motor de búsqueda devuelve tripletas para cada una de las consultas en el orden determinado por el valor de ranking. Para presentar la información al usuario se agrupan las tripletas según la oración en la que aparecen.

¿Qué ley crea la bandera de Salta?
Pron.Interr.: qué Verbo: crear Antes de Verbo:[ley] Dsp. de Verbo:[bandera, Salta]
(+arg1:ley +arg2:bandera +arg2:salta +rel: crear)
Tripleta 1: 0.26398262 banderadesalta.txt: -arg1 Mediante la ley N 6.946 -arg2 1996 la Bandera de Salta -rel crear

Figura 3. Ejemplo 1

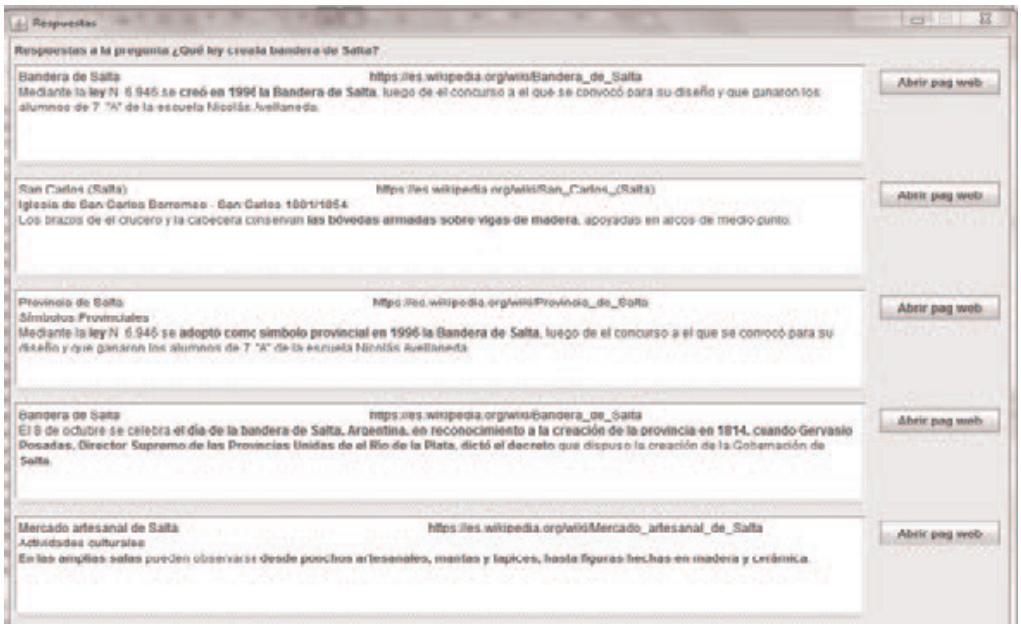

Figura 4. Interfaz del sistema de búsqueda de respuestas, Ejemplo 1

Cada oración se muestra en un panel junto a un botón que lleva al usuario a la página web en la que aparece la oración en su contexto. En la Figura 4 se muestran las cinco primeras respuestas obtenidas para la pregunta del ejemplo anterior. Nótese que la primera y tercera respuesta son parecidas, pero provienen de documentos diferentes y por ello se muestran dos veces. Nótese que la segunda y quinta respuestas no se corresponden a la información buscada. Esto se debe a que algunas de las heurísticas, al aplicar lematización, ampliaron la búsqueda a palabras similares a las de la consulta. Por ejemplo,

(+arg1:ley~2+tituloDoc:salta+arg2: bandera~2 + rel : (crear armar...)

fue una de las cláusulas conjuntivas creadas (añadiendo además el verbo armar como sinónimo de crear con la heurística iv). Con esta cláusula Lucene devuelve la siguiente tripleta correspondiente a la segunda oración de la respuesta:

Tripleta 22: 0.086254016 sancarlossalta.txt:

-arg1 las bóvedas

-arg2 sobre vigas de madera

-rel armar

Revista Tecnología y Ciencia
ISSN 1666-6933

Ajustando las heurísticas o la manera de presentar las respuestas al usuario, o incluso permitiendo solo tripletas por encima de un umbral de ranking, podrían descartarse las respuestas no relevantes, aumentando así la precisión del sistema, pero a riesgo de disminuir el recall, es decir, de no encontrar respuestas que sí son relevantes. En base a nuestros experimentos decidimos permitir una reducción en precisión para aumentar el recall.

La cuarta respuesta muestra una de las limitaciones del enfoque de nuestro sistema, basado en un análisis superficial del texto y no en un análisis sintáctico y semántico profundo. La pregunta es "qué ley" pero la respuesta no se refiere a la ley sino al día de celebración de la bandera.

En el Ejemplo 2 (Figura 5. Ejemplo 2) se aplica la heurística (vii) en que solo un argumento, en este caso arg2, es presentado en la consulta.

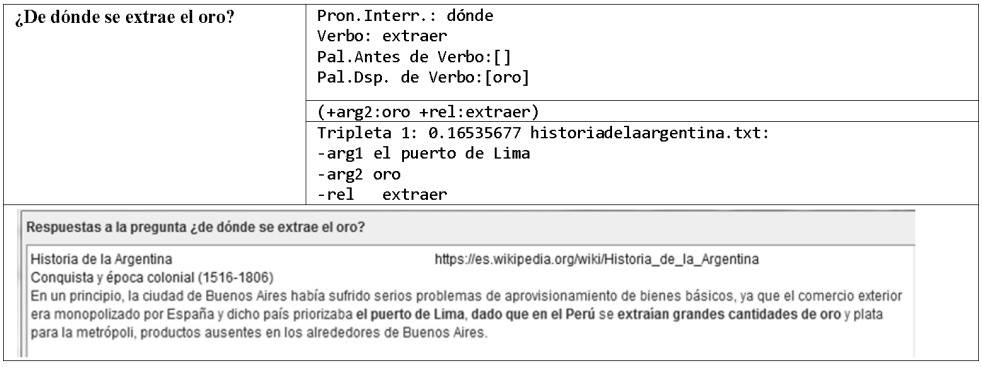

Figura 5. Ejemplo 2

El Ejemplo 3 (Figura 6) muestra una combinación de las heurísticas (ii), (iii), y (iv). Las palabras en plural en la pregunta se lematizan. Así por ejemplo +arg1:chiriguano~2 permite una equiparación aproximada, las palabras que aparecen después del verbo se colocan en diversos lugares de los argumentos de las tripletas, y además del verbo tener se añaden sus sinónimos.

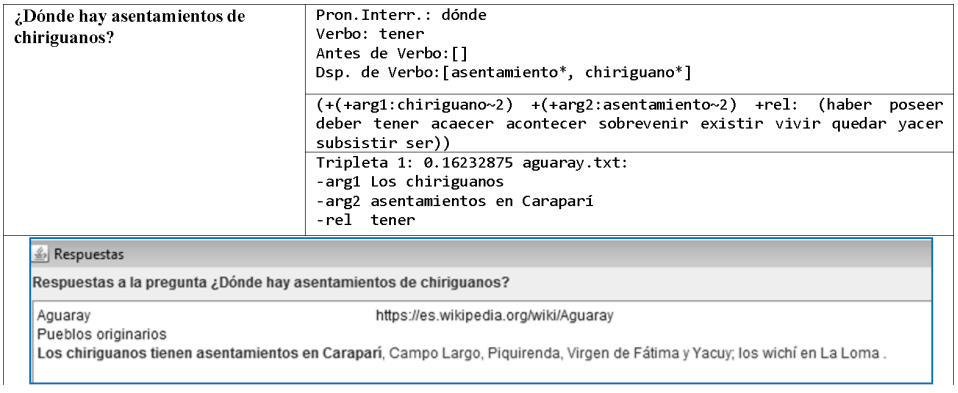

Figura 6. Ejemplo 3

Experimentos

Como ya se ha mencionado, el enfoque de este trabajo sobre sistemas de búsqueda de respuestas es la extracción de relaciones, creando una base de conocimientos a partir de la cual se obtienen las respuestas. La evaluación del enfoque propuesto tiene por tanto dos aspectos:

a) evaluación del componente de extracción de relaciones

b) evaluación del sistema completo de búsqueda de respuestas

Evaluación del componente de extracción de relaciones

El componente de extracción de relaciones está inspirado en los propuestos por ReVerb (Fader, Soderland, & Etzioni, 2011) y ExtrHech (Zhila & Gelbukh, 2013), una aplicación de ReVerb para el idioma español. Por tanto para la evaluación del componente de extracción de relaciones parece apropiado compararlo con ExtrHech.

ExtrHech utiliza Freeling para generar etiquetas POS siguiendo el estándar EAGLES. Para extraer tripletas define la frase verbal como (V W*P | V) donde V puede ser un verbo simple, posiblemente precedido por un pronombre reflexivo, o un participio, W puede ser un nombre, adjetivo, adverbio, pronombre o artículo, y P una preposición inmediatamente seguida de un infinitivo. En (Zhila & Gelbukh, 2013) ExtrHech es evaluado sobre un corpus de 68 oraciones gramatical y ortográficamente correctas seleccionadas de libros de texto.

Para evaluar el sistema se tienen en cuenta dos métricas diferentes pero relacionadas:

precisión: porcentaje de tripletas extraídas que son correctas

recall: porcentaje de extracciones obtenidas del total de posibles extracciones correctas.

El conjunto de posibles extracciones correctas está formado por las tripletas correctas obtenidas por el sistema más otras tripletas manualmente extraídas que se espera que el sistema sea capaz de encontrar.

Los autores de ExtrHech reportan una precisión del 87% y un recall del 70%.

Hemos evaluado nuestro componente de extracción de relaciones con ese mismo conjunto de 68 oraciones. El sistema obtuvo 347 tripletas. De ellas se consideran correctas 239, lo cual lleva a una precisión del 69%. Para calcular el recall, se consideró que el número de tripletas que se podrían obtener de las 68 oraciones a nuestro criterio es 267; de ellas son las 239 obtenidas correctamente y otras 28 que nuestro sistema no logra obtener. En base a ello el recall es el 90%.

La Tabla 4 muestra un resumen de los resultados. La métrica F1 combina precisión y recall y es frecuentemente usada en sistemas de recuperación de la información.

Nótese que nuestro sistema (QA) está sesgado hacia mayor recall ya que el objetivo final de la extracción de relaciones es generar una base de conocimientos para

Revista Tecnología y Ciencia
ISSN 1666-6933

contestar preguntas. Cuanto mayor sea esa base de conocimientos creada para responder preguntas, mayor posibilidad de encontrar respuestas. Un análisis de las tripletas incorrectas indica que en general no producen información errónea de utilizarse para responder preguntas factuales, sino que son incongruentes o incompletas. El sesgo de nuestro sistema hace que de una misma oración se extraigan una variedad de tripletas, por lo que en general la tripleta o tripletas correctas también son extraídas junto con las incorrectas. A la hora de responder una pregunta del usuario, esta se presenta como una consulta de Lucene al motor de búsqueda en la base de tripletas, por lo que las tripleta incompleta o incongruentes no son las que el motor de búsqueda devuelve usualmente en primer lugar. Además, como se verá más adelante, las tripletas obtenidas como respuesta se muestran al usuario en su contexto, que es la oración de la que proceden, por lo que en caso de ser incongruentes o incompletas el contexto proporciona la información que puede faltar.

En corpus de tamaños masivos, como puede ser toda la web, son necesarios mecanismos para disminuir el número de tripletas sin perder precisión, pero probablemente a costa de recall. Esta es la utilidad de la restricción léxica de Reverb que usa estadísticas sobre el uso de las tripletas para evitar tripletas demasiado específicas en la base de conocimientos. Esto no es necesario en nuestro caso.

	precisión	recall	F1
ExtrHech	87%	70%	0.776
QA	69%	90%	0.779

Tabla 4. Comparación de los resultados para la extracción de relaciones

Construcción del corpus

Para la evaluación del sistema completo de búsqueda de respuestas se preparó un corpus de documentos obtenidos de la web usando técnicas de web scraping. Tras una limpieza inicial, principalmente eliminando etiquetas HTML y secciones irrelevantes de las páginas, el contenido de las mismas se almacenó en archivos de texto junto con metadatos tales como el título de la página, la URL, y la división del texto en secciones con sus correspondientes títulos.

El contenido de los documentos es información sobre la Provincia de Salta. Se han obtenido colecciones de archivos provenientes de distintos sitios web:

- turismo.salta.gov.ar
- salta.gov.ar.
- Wikipedia

Evaluación del sistema de búsqueda de respuestas

A partir del corpus de documentos, se obtuvieron 456.765 tripletas. Éstas se indexaron para la construcción de un motor de búsqueda con Lucene.

Para evaluar el sistema de búsqueda de respuestas a preguntas en lenguaje natural usando esas tripletas se creó un banco de 150 preguntas relacionadas con la Provincia de Salta. Todas las preguntas tienen respuestas dentro de los documentos del corpus.

Para tener algún punto de comparación se propusieron las mismas preguntas al motor de búsqueda de Google. Google dispone de la información de Wikipedia y del sitio oficial de la Provincia de Salta, de la cual nuestro corpus es un subconjunto, así como de otras muchas páginas de temáticas relacionadas. Es de esperar que por tanto la búsqueda en Google brinde documentos que respondan a todas las preguntas. Por otro lado, el objetivo es la obtención de respuestas concretas, no de documentos donde posiblemente esté la respuesta. Es éste el componente en el que el sistema de búsqueda de respuestas se ha comparado con el buscador de Google.

La Tabla 5 resume la evaluación. Las columnas corresponden al número de preguntas que fueron respondidas o no por nuestro sistema (QA). Por respondidas se entiende que una respuesta adecuada aparece entre las cinco primeras respuestas. Las filas de la Tabla 5 se refieren al número de preguntas respondidas por Google (G). Solo se han considerado las cinco primeras respuestas de dicho buscador. Las respuestas pueden aparecer de manera explícita (por ejemplo en un recuadro, o en el breve resumen de la página que devuelve el buscador) o bien estar contenidas en uno de los vínculos propuestos. En este caso el usuario debe acceder a la página y buscar en ella la respuesta. La segunda sección de la Tabla muestra los mismos datos en porcentaje del total de preguntas.

	QA si	QA no	Todas
G explícita	78	7	85
G en página	51	10	61
G no	3	1	4
Todas	132	18	

	QA si	QA no	Todas
G explícita	0.52	0.05	0.57
G en página	0.34	0.07	0.41
G no	0.02	0.01	0.03
Todas	0.88	0.12	

Tabla 5. Resultados de la evaluación

De las 150 preguntas, Google mostró una respuesta explícita en 57% de las preguntas y una página que contiene la respuesta en 41% de los casos. En cuatro casos (3%) no devolvió una respuesta correcta. El sistema QA devolvió una respuesta en el 88% de los casos y devolvió respuestas inadecuadas en el 12%. Nótese que Google tiene a su disposición más fuentes de datos que QA, pero en 89% de los casos las respuestas de Google sobre Salta se extrajeron de Wikipedia, o de turismosalta.gob.

Revista Tecnología y Ciencia
ISSN 1666-6933

ar, las mismas fuentes que QA utiliza .

En el 76% de los casos QA mostró la(una) respuesta correcta en primer lugar, en el 8% de los casos la respuesta apareció en segundo lugar, y en 3% y 1% de los casos en tercer y cuarto lugar. En el 12% no hubo una respuesta adecuada.

Las razones más frecuentes de los errores son las siguientes:

- Una palabra de la pregunta es requerida en la consulta pero no aparece en el documento y por tanto no está en la tripleta (6 de 18 errores)
- La consulta obtenida tras el análisis de la pregunta es demasiado amplia y la respuesta correcta no aparece entre las cinco mostradas (5 de 18 errores)
- El tiempo verbal no es considerado y las respuestas no son válidas (2 de 18 errores). Por ejemplo a la pregunta ¿Cuál es la capital de la Argentina? se obtienen diversas respuestas sobre la capital en el pasado, que superan el límite de cinco respuestas, entre las cuales no está la respuesta correcta.

A partir de este análisis de errores surgen algunas posibilidades de mejora de las estrategias e heurísticas utilizadas:

- para el primer tipo de error, el más frecuente, en que hay palabras de la pregunta que no aparecen en el documento, se pueden incorporar sinónimos de nombres comunes o adjetivos. La utilización de ontologías o bases de conocimiento con este objeto, como Wordnet, es posible, aunque no existen tantos recursos lingüísticos adecuados en castellano como en inglés.
- para focalizar la respuesta se puede utilizar el tipo de pregunta y el tipo de respuesta esperada para el mismo, incorporando el análisis de las entidades con nombre. Las respuestas propuestas por Lucene pueden ser filtradas para limitarlas a tripletas que contengan una entidad del tipo esperado en la respuesta.
- incorporando tripletas no basadas en un verbo se podrían responder preguntas para las que la respuesta no aparece en una oración. Por ejemplo la respuesta a la pregunta ¿En qué año falleció Artidorio Cresseri? aparece en una página de Wikipedia de la siguiente manera:

  "Artidorio Cresseri (Salta, 5 de marzo de 1862 -
  ibídem, 18 de octubre de 1950)"

Este patrón podría implementarse para extraer tripletas con primer argumento Artidorio Cresseri, y segundo argumento una fecha, e incluso a incorporar en esas tripletas las relaciones nació y falleció.

Conclusiones

Las técnicas de extracción de relaciones son un potente instrumento para extraer conocimiento de grandes volúmenes de texto. Además de para responder preguntas, las relaciones también pueden utilizarse para poblar ontologías, y para

el llenado de plantillas (template filling) para situaciones estereotípicas extrayendo del texto la información y estructurándola en la plantilla. Una aplicación de las plantillas es la generación de escenas (visuales) correspondientes al conocimiento capturado en el texto. La extracción de relaciones se ha aplicado también a textual entailment. Otras aplicaciones interesantes son, a partir de la gran cantidad de literatura biomédica disponible, descubrir interacciones entre proteínas (que aparecerían como relaciones en el texto), o relaciones entre genes y enfermedades. Por tanto el desarrollo de nuevas aplicaciones de la extracción de relaciones es una línea abierta en el estado del arte de la minería de textos y el procesamiento del lenguaje natural.

Este trabajo ha demostrado la aplicación exitosa de las técnicas de extracción de relaciones al problema de la búsqueda de respuestas. Para ello se ha construido una base de conocimientos en forma de tripletas basadas en un verbo, que describen relaciones semánticas entre los objetos que aparecen en el texto y facilitan su comprensión. El enfoque del trabajo ha hecho uso de técnicas eficientes de análisis superficial del texto y el sistema construido tiene una precisión y recall comparable a otros sistemas del estado del arte. Los resultados de este sistema de búsqueda de respuestas sobre un banco de preguntas a un corpus de 2052 documentos sobre Salta obtenidos de la web demuestran la validez de este enfoque.

Revista Tecnología y Ciencia
ISSN 1666-6933

Referencias

Cardoso, A. C., Bini, A., y Pérez Abelleira, M. A. (2015). "Diseño de un sistema de búsqueda de respuestas para diversos tipos de preguntas". Simposio Argentino de Inteligencia Artificial, 44° Jornadas Argentinas de Informática (JAIIO). Rosario, Santa Fe: 25-32.

Carlson, P. Apache Lucene Query Parser Syntax. Obtenido de <https://lucene.apache.org/core/2_9_4/queryparsersyntax.pdf> (21 de 06 de 2013).

Etzioni, O., Banko, M., y Cafarella, M. (2006). "Machine Reading". AAAI'06 Proceedings of the 21st National Conference on Artificial intelligence. AAAI Press: 1517-1519.

Fader, A., Soderland, S., y Etzioni, O. (2011). "Identifying Relations for Open Information Extraction". EMNLP '11 Proceedings of the Conference on Empirical Methods en Natural Language Processing. Stroudsburg, PA: Association for Computational Linguistics: 1535-1545.

Ferrucci, D., y Lally, A. (2004). "Building an example application with the Unstructured Information Management Architecture" en *IBM Systems Journal*, 45(3).

Gamallo, P. (2014). "An Overview of Open Information Extraction". 3rd Symposium on Languages, Applications and Technologies (SLATE'14). Alemania: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing: 13-16.

Mausam, Schmitz, M., Bart, R., Soderland, S., y Etzioni, O. (2012). "Open Language Learning for Information Extraction". Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning: 523-534.

Wikipedia (2015). Gráfico de conocimiento - Wikipedia, La enciclopedia libre. Recuperado el 8 de Marzo de 2016, de <https://es.wikipedia.org/w/index.php?title=Gr%C3%A1fico_de_conocimiento&oldid=82616528>.

Wu, F., y Weld, D. (2010). "Open Information Extraction using Wikipedia". ACL '10 Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. Upsala: Association for Computational Linguistics: 118-127.

Zhila, A., y Gelbukh, A. (2013). !Comparison of open information extraction for English and Spanish!. Proceedings Dialogue '2013.

Agradecimientos

Este trabajo ha sido financiado en parte por el Consejo de Investigaciones de la Universidad Católica de Salta (Resol Rect 839/13).