Extracción de relaciones para la búsqueda automática de respuestas

Alejandra Carolina Cardoso, Enzo Notario, M. Alicia Pérez Abelleira[1]
[1] Facultad de Ingeniería e Informática e IESIING. Universidad Católica de Salta
Campo Castañares s/n, A4400 Salta, Argentina
{acardoso, aperez}@ucasal.edu.ar, enzo.notario@gmail.com

Resumen. El presente trabajo explora el problema de la búsqueda automática de respuestas en grandes cantidades de texto como una aplicación del problema de extracción de relaciones. Las relaciones extraídas, en forma de tripletas basadas en un verbo, describen relaciones semánticas entre los objetos que aparecen en el texto y facilitan su comprensión. Las tripletas extraídas forman la base de conocimientos sobre la que se hacen consultas para obtener respuestas a preguntas en lenguaje natural. 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. El sistema de búsqueda de respuestas se evaluó sobre un banco de preguntas a un corpus de 2052 documentos sobre Salta obtenidos de la web.

Keywords: búsqueda de respuestas, extracción de relaciones, UIMA.

1 Introducción

La búsqueda de respuestas tiene como objetivo dar respuestas en lenguaje natural a preguntas también en lenguaje natural. Estas respuestas se buscan en grandes cantidades de texto, y la disponibilidad cada vez mayor de grandes bases de documentos, incluida la web, aumenta la importancia e interés de este aspecto de la minería de textos. El presente trabajo explora el problema de la búsqueda de respuestas 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.

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 dicha tarea al problema de búsqueda de respuestas, seguido de su evaluación experimental en la Sección 5 y algunas conclusiones.

2 Extracción de relaciones

En los últimos años ha habido mucho interés en la posibilidad de extraer grandes cantidades de proposiciones básicas de grandes volúmenes de texto. Esta tarea 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 particular entre dos o más entidades de un texto [1].

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. 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, detectando 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 [2] [3]. Estos sistemas detectan demasiadas relaciones que son incoherentes, debido a que el extractor primero detecta los argumentos y luego busca la relación entre ellos [4].

Análisis del contexto, en que se extraen relaciones no solo centradas en un verbo, sino otros tipos de relaciones binarias, e incluso con más de dos argumentos [5]. 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, un enfoque intermedio entre los anteriores. Las relaciones están centradas en el verbo pero 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 [4] y ExtrHech [6]. Nuestro sistema está basado en este tercer enfoque.

3 Arquitectura del sistema

El problema de búsqueda de respuestas presenta tres fases:

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

b) extracción e indexación de las tripletas usando un motor de búsqueda. En este trabajo se usó Lucene.

c) búsqueda de respuesta: 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 documento se usan como contexto para presentar la respuesta al usuario.

Para la implementación y experimentos del trabajo se utilizó la arquitectura UIMA (*Unstructured Information Management Architecture*) [7] 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 Fig.1 muestra el proceso de extracción de relaciones de un corpus de documentos de texto, correspondiente a las fases (a) y (b).

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

Tras la separación del texto en oraciones, un anotador POS basado en FreeLing[1] realiza el análisis morfológico de cada oración obteniendo para cada palabra su lema y etiqueta EAGLES[2], que identifica cada palabra como artículo, verbo, adjetivo, etc. En este trabajo son suficientes los primeros caracteres de la etiqueta EAGLES (categoría y tipo) para identificar cada palabra. La Tabla 1 (segunda fila) muestra las etiquetas obtenidas en el análisis de una oración de ejemplo.

Tabla 1. Etiquetas POS y *chunks* para una oración de ejemplo

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

A partir de la secuencia de etiquetas POS la oración se divide en fragmentos más grandes, o *chunks*, centrados en un verbo (VP) o en un nombre (NP). La etiqueta O se utiliza para los elementos de la oración que no forman parte de un *chunk*, tales como pronombres interrogativos o signos de puntuación. La tercera fila de la Tabla 1 muestra los *chunks* obtenidos.

El anotador que extrae relaciones en forma de tripletas recibe la descomposición en *chunks* y las etiquetas POS y comienza extrayendo la relación, centrada en torno al verbo utilizando el patrón de la Fig. 2. Este patrón es una extensión de la restricción sintáctica utilizada por ReVerb [4].

La frase verbal *rel* se almacena en la tripleta con el verbo en infinitivo y eliminando los pronombres (*se*, *le*, etc.). A continuación se localizan los argumentos. Para cada frase verbal *rel* obtenida se buscan *chunks* a la izquierda y a la derecha de la frase verbal (*arg1* e *arg2* respectivamente). De esta forma se obtiene la primera tripleta candidata <*arg1*, *rel*, *arg2*>, que es la tripleta con la mayor cantidad de palabras. Si cada argumento contiene por lo menos un nombre común o nombre propio, la nueva variante se agrega al conjunto de tripletas.

[1] http://nlp.lsi.upc.edu/freeling/
[2] http://nlp.lsi.upc.edu/freeling/doc/tagsets/tagset-es.html

V | *VW*P* | *VW*P+*

donde: *V* = RN? P0? VA? (VM| VS |VP) (VN| VP)?
W = (NC | NP| AQ | AO | RG | PP | PD| PX| PI |DA)
P = SP (VN | VG)?

y: * representa repetición cero o más veces
+ representa repetición una o más veces
? representa repetición cero o una vez

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

De esta forma se genera un gran número de tripletas, algunas de ellas redundantes, incoherentes o hasta incorrectas. Sin embargo, dada la aplicación de las tripletas a responder preguntas, se prefiere un gran número de tripletas para aumentar el *recall* aún a riesgo de disminuir la precisión.

Finalmente el conjunto de tripletas se indexa para una búsqueda eficiente usando un motor de búsqueda basado en Lucene. La unidad de texto indexada es la tripleta a la que se añaden la oración de la que extrajo y los datos del documento en el que aparece la oración.

4 Búsqueda de respuestas

Como ya se indicó, este trabajo encara el problema de búsqueda de respuestas como búsqueda en una base de conocimientos de relaciones. Por tanto, una pregunta planteada por un usuario en lenguaje natural se convierte en una consulta adecuada para el motor de búsqueda en el conjunto de relaciones. El proceso tiene similitudes con el descripto para la extracción de tripletas. Se realiza el etiquetado POS, se ubican los marcadores de la pregunta (pronombres interrogativos, ya que se trata de preguntas factuales) y se detecta el verbo y los *chunks* nominales que lo preceden o siguen. Puede haber cero, 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. Cada heurística reduce las restricciones en la consulta, especialmente si no se han encontrado respuestas a las versiones anteriores de la consulta. Por ejemplo, la primera heurística consiste en buscar los elementos de la pregunta (verbo, chunks antes y después del verbo) en los elementos (verbo, *arg1*, *arg2*) de la tripleta. Si no se obtienen resultados, la consulta se amplía a sinónimos del verbo. Al ampliar la consulta con ésta y otras heurísticas, aumenta el *recall* a riesgo de disminuir la precisión e introducir respuestas que no son relevantes a la consulta.

El resultado de este proceso es un conjunto de tripletas que responden a la consulta, ordenadas según su relevancia a la misma. El usuario obtiene las oraciones correspondientes a las tripletas devueltas.

Para la pregunta del Ejemplo 1 (Fig. 3) se observa la aplicación de la primera heurística. La palabra *ley* que aparece antes del verbo se busca en el campo *arg1* de la tripleta, las palabras del *chunk* que aparece después del verbo se buscan en el campo *arg2*, y el verbo en el campo *rel*. El motor de búsqueda devuelve tripletas para las consultas generadas por las heurísticas. Estas tripletas se agrupan según la oración en la que aparecen y se seleccionan las primeras cinco oraciones, que se muestran al usuario. En cada oración se resaltan los elementos que forman la tripleta para destacar la posible respuesta en su contexto. La Fig. 3 muestra la primera respuesta obtenida.

Pregunta: *¿Qué ley crea la bandera de Salta?*	
Pronombre Interrogativo: qué Verbo: crear Antes de Verbo:[ley] Después de Verbo:[bandera, Salta]	Consulta Lucene: (+*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	

Fig. 3 Ejemplo 1

En el Ejemplo 2 (Fig. 4), se utilizan otras heurísticas: lematizar las palabras en plural de la pregunta (+*arg1:chiriguano~2* permite una equiparación aproximada), buscar sinónimos del verbo, y cambiar el orden de los argumentos de la tripleta.

Pregunta: *¿Dónde hay asentamientos de chiriguanos?*	
Pronombre Interrogativo: dónde Verbo: tener Antes de Verbo:[] Después de Verbo:[asentamiento*, chiriguano*]	Consulta Lucene (+*arg1*:chiriguano~2 +*arg2*:asentamiento~2 +*rel*: (haber poseer deber tener acaecer acontecer sobrevenir existir vivir quedar yacer subsistir ser))
Tripleta 1: 0.16232875 aguaray.txt: -*arg1:* Los chiriguanos -*arg2:* asentamientos en Caraparí -*rel:* tener	

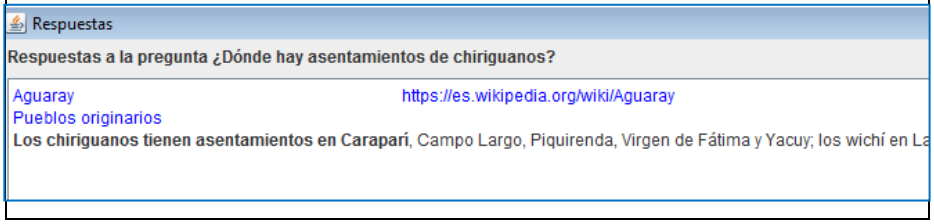

Fig. 4 Ejemplo 2

5 Experimentos

La evaluación del enfoque propuesto tiene dos aspectos:

a) evaluación del componente de extracción de relaciones
b) evaluación del sistema completo de búsqueda de respuestas.

El componente de extracción de relaciones está inspirado en los propuestos por ReVerb [4] y ExtrHech [6], una aplicación de ReVerb para el idioma español. Por tanto para la evaluación del componente de extracción de relaciones parece apropiado compararlo con ExtrHech. En [6], ExtrHech es evaluado sobre un corpus de 68 oraciones gramatical y ortográficamente correctas seleccionadas de libros de texto. 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 obteniendo 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%.

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

Para la evaluación del sistema completo de búsqueda de respuestas, se preparó un corpus de 2052 documentos sobre la Provincia de Salta obtenidos de la web[3] usando técnicas de *web scraping*. Se eliminaron las 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. La extracción de los diversos elementos se realizó mediante expresiones regulares. A partir del corpus de documentos, se obtuvieron 456.765 tripletas. Se creó un banco de 150 preguntas relacionadas con la Provincia de Salta para evaluar el sistema. 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 que 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

[3] turismo.salta.gov.ar, Wikipedia

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.

Tabla 2. Resultados de la evaluación

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

	QA sí	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	

La Tabla 2 resume la evaluación. En la sección izquierda, 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 mostradas al usuario. Las filas 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 tercera fila refleja los casos en que Google no obtuvo una respuesta correcta. La sección de la derecha muestra los mismos resultados en porcentaje.

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. Nuestro sistema devolvió una respuesta en el 88% de los casos. (De estos, en el 76% de los casos se obtuvo 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. Nótese que Google tiene a su disposición gran variedad de fuentes de datos, pero el 89% de las respuestas de Google sobre Salta se extrajeron de las mismas fuentes que se utilizaron para construir el corpus[4].

Las razones más frecuentes de los errores de nuestro sistema 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.

[4] A lo largo del presente proyecto Google ha mejorado su comprensión de Wikipedia, y es uno de los sitios privilegiados en los que busca información, parte del proyecto Knowledge Graph, "una base de conocimiento usada por Google para mejorar los resultados obtenidos con su motor de búsqueda mediante información de búsqueda semántica recolectada de una amplia gama de recursos" [8]. Por ello estimamos que los resultados del sistema hace unos meses se habrían comparado con Google más favorablemente que ahora.

6 Conclusiones

Las técnicas de extracción de relaciones son un potente instrumento para extraer conocimiento de grandes volúmenes de texto. Una de sus aplicaciones es la búsqueda de respuestas, como se ha demostrado en este trabajo. 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.

Referencias

1. Michelle Banko and Oren Etzioni, "The Tradeoffs Between Open and Traditional Relation Extraction," in Proceedings of ACL-08: HLT, Columbus, Ohio, 2008, pp. 28-36.
2. Oren Etzioni, Michelle Banko, and Michael J. Cafarella, "Machine Reading," in AAAI'06 Proceedings of the 21st National Conference on Artificial intelligence, 2006, pp. 1517-1519.
3. Fei Wu and Dan Weld, "Open Information Extraction using Wikipedia," in ACL '10 Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Upsala, 2010, pp. 118-127.
4. Anthony Fader, Stephen Soderland, and Oren Etzioni, "Identifying Relations for Open Information Extraction," in EMNLP '11 Proceedings of the Conference on Empirical Methods in Natural Language Processing, Stroudsburg, PA, 2011, pp. 1535-1545.
5. Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni, "Open Language Learning for Information Extraction," in Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 2012, pp. 523-534.
6. A. Zhila and A. Gelbukh, "Comparison of open information extraction for English and Spanish," in Proceedings Dialogue'2013, 2013.
7. D. Ferrucci and A. Lally, "Building an example application with the Unstructured Information Management Architecture," IBM Systems Journal, vol. 45, no. 3, 2004.
8. Wikipedia. (2015) Gráfico de conocimiento --- Wikipedia, La enciclopedia libre. [Online]. https://es.wikipedia.org/w/index.php?title=Gr%C3%A1fico_de_conocimiento&oldid=82616528

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