Ask Classora!: Fuentes ∗structuradas, Preguntas no Estructuradas, Respuestas Estructuradas

Yerai Doval[1], Jesús Vilares[1], Carlos Gómez-Rodríguez[1], and Iván Gómez-Caderno[2]

[1] Grupo LYS, Departamento de Computación, Facultade de Informática, Universidade da Coruña, Campus de A Coruña, 15071 – A Coruña
{yerai.doval, jvilares, cgomezr}@udc.es – www.grupolys.org

[2] Classora Technologies S.L., Ronda de Outeiro 116, 15008 – A Coruña
ivan.caderno@classora.com – http://www.classora-technologies.com

Resumen Presentamos Ask Classora!, una interfaz en lenguaje natural para la Classora Knowledge Base, base de conocimiento comercial alimentada a partir de fuentes web. La interfaz permite al usuario interaccionar con dicha base en su propio idioma, de forma que sus consultas sean interpretadas y traducidas al lenguaje formal de consulta para la interrogación del sistema. Los resultados obtenidos son además enriquecidos por nuestra herramienta. El objetivo es comprobar la validez de este tipo de soluciones de cara a una futura puesta en explotación.

Keywords: Base de conocimiento online, interfaz de lenguaje natural, lenguaje formal de consulta, español.

1. Introducción

A día de hoy, *Classora Knowledge Base* (CKB),[3] o *Classora*, es la mayor base de conocimiento comercial *online* en español. Desarrollada por Classora Technologies,[4] su objetivo es integrar información procedente tanto de fuentes web públicas como privadas (Wikipedia, INE, FMI, etc.) como de usuarios, enriqueciendo el resultado con valores añadidos tales como su estructuración y adición de metadatos explicativos o la disponibilidad de herramientas para su presentación en diversos formatos (tablas, gráficas, mapas, etc.) y análisis OLAP.

A la hora de interrogar al sistema, sus creadores desarrollaron un lenguaje formal de consulta, de naturaleza sencilla: el *Classora Query Language* (CQL). Sin embargo, durante posteriores sondeos de mercado con vistas a la explotación comercial de CKB, la firma detectó una demanda común a buena parte de sus clientes potenciales: posibilitar un acceso más natural e intuitivo a los datos, de modo que se libere al usuario de la necesidad de aprender el lenguaje formal de consulta de la herramienta. Es en este contexto cuando se plantea la necesidad de desarrollar un prototipo de *interfaz web de lenguaje natural* en español a

[3] http://www.classora.com
[4] http://www.classora-technologies.com

modo de demostrador tecnológico, de cara a permitir la interacción con la base de conocimiento online mediante consultas en lenguaje natural. Dicha interfaz se encargaría de traducir una consulta expresada en lenguaje natural a CQL.

En este trabajo, tras presentar la CKB y los mecanismos empleados para su población a partir de fuentes web diversas, nos centraremos en *Ask Classora!*, una interfaz de lenguaje natural desarrollada para su interrogación por parte de los usuarios y el enriquecimiento de resultados.

A partir de aquí, la Sección 2 presenta al lector la base de conocimiento y sus mecanismos de población. Seguidamente, la Sección 3 nos introduce en las interfaces de lenguaje natural. La Sección 4 presenta de forma general nuestra interfaz y su contexto. El traductor del sistema, su *core*, es descrito en detalle en la Sección 5, mientras que los mecanismos de enriquecimiento de resultados son presentados en la Sección 6. Nuestro prototipo es evaluado en la Sección 7 para, finalmente, presentar nuestras conclusiones y desarrollo futuro en la Sección 8.

2. Classora Knowledge Base

La información almacenada en *Classora* se estructura en torno a los siguientes *conceptos*:

- *Unidad de conocimiento*: tipo de entidad sobre la que se guarda conocimiento (p.ej. persona, empresa, país, etc).
- *Atributos*: conocimiento que se guarda sobre cada tipo de entidad (p.ej. para persona: edad, fecha de nacimiento, etc).

La Figura 1 muestra un esquema general de la arquitectura del sistema. Como ya hemos apuntado, éste es alimentado mediante diversas fuentes web, tanto públicas (Wikipedia, INE, Banco Mundial, FMI, etc.) como privadas, siendo potencialmente aplicable a toda información de interés disponible en la Red. Las fuentes pueden ser tanto estructuradas como semi-estructuradas o no-estructuradas,[5] y a la hora de procesarlas para agregar al sistema la información que contienen, éste emplea lo que se denomina el *módulo de extracción de datos*, que consiste en un conjunto de robots coordinados (*crawlers*) de tres tipos:

1. *Robots de extracción.* Realizan las cargas masivas de informes a partir de fuentes públicas oficiales (CIA, FMI, Eurostat, etc.), tanto *absolutas* —se extrae toda la información de una determinada fuente; como *incrementales* —se monitorizan las fuentes en busca de cambios para trasladarlos al sistema.
2. *Exploradores de datos.* Buscan y actualizan datos concretos de una unidad de conocimiento a partir de fuentes concretas (Wikipedia, Banco Mundial, etc.)
3. *Agregadores de contenidos.* Éstos no se conectan a fuentes externas, sino que usan los datos ya almacenados para generar nueva información, no siempre evidente, empleando técnicas de *Data mining* y OLAP.

Además de estos procedimientos automáticos, es posible introducir información manualmente (usuarios y grupos de generación de contenidos).

[5] De ahí el juego de palabras del título, al emplear el comodín $*$.

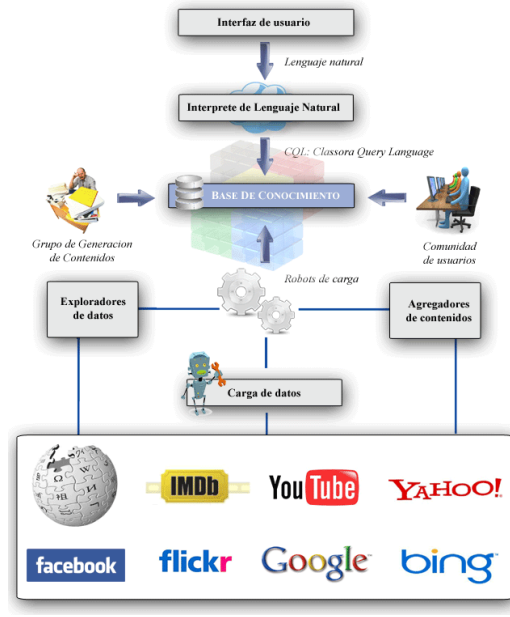

Figura 1. Arquitectura general de Classora.

3. Interfaces de Lenguaje Natural

Al interaccionar con repositorios de información como bases de conocimiento o bases de datos, es habitual emplear *lenguajes formales de consulta*; por ejemplo:

SQL:
```
SELECT nombre FROM persona
    WHERE lugar_de_nacimiento='España' && edad>50;
```

CQL:
```
persona*, lugar_de_nacimiento='España', edad>50
```

Sin embargo, esto no resulta intuitivo ni práctico, sobre todo para usuarios inexpertos, que preferirían formular la consulta en su propio idioma:

"¿Qué españoles tienen más de 50 años?"

Aunque el usuario podría intentar aprender a usar el lenguaje de consulta, dominarlo no es trivial e implica un período de formación. Por eso se suele recurrir a algún tipo de *interfaz de consulta* gráfica o basada en formularios [1,12], lo que no siempre es suficiente pues éstos pueden también requerir de un período de aprendizaje. Además, tal y como se desprende de [14], los usuarios, sobre todo los inexpertos, prefieren usar interfaces en lenguaje natural basados en consultas en forma de enunciado a aquéllas que emplean palabras clave, y a su vez las palabras clave a los menús o interfaces gráficas [12, Cap. 4]. Por otra parte, este tipo de interfaces son, por lo general, limitadas y poco flexibles, limitaciones especialmente relevantes a la hora de operar sobre datos semánticamente enriquecidos como es nuestro caso. Esto se debe a que si bien estos sistemas son

capaces de dar solución a consultas mucho más complejas que los sistemas clásicos, formular ese tipo de consultas empleando dichas interfaces, palabras clave o incluso directamente mediante un lenguaje formal de consulta, puede resultar excesivamente complejo o incluso ser imposible [8,1].

Como respuesta a esta problemática, surgen entonces las *Interfaces de Lenguaje Natural* (ILN) [8,1], las cuales permiten superar tales limitaciones, de forma que el usuario pueda interactuar de manera efectiva con el sistema de un modo sencillo y natural empleando para ello su propia lengua.

3.1. Trabajos Relacionados

Se puede constatar una evolución en la investigación y aplicación de las ILN para sistemas de información. Inicialmente, a principios de los 70 se centran en las bases de datos. Por aquel entonces destaca LUNAR [23], para consultas sobre una base de datos de análisis químicos de rocas lunares. Poco después aparece RENDEZVOUS [7], capaz de cierta interacción a la hora de resolver algunas ambigüedades. A principios de los 80 surge CHAT-80 [22], que emplea un inglés acotado para interrogar una base de datos Prolog geográfica. Poco después, sin embargo, las ILN para bases de datos pasan a segundo plano ante el poco interés comercial despertado, motivado éste por la disponibilidad de alternativas más sencillas basadas en interfaces gráficas o formularios [1,12].

Ya en los 90, la popularización de la Web y los motores de búsqueda atrajo de nuevo el interés por la investigación en las ILN, esta vez como alternativa al uso de consultas basadas en palabras clave. Sin embargo, tampoco esta vez es correspondido por un interés comercial [12, Cap. 4].

Actualmente existe un interés creciente por las ILN para ontologías y su aplicación a Web Semántica y *Linked Data*. Entre ellas destacamos los siguientes sistemas para idioma inglés, de propósito similar al nuestro:

- ORAKEL [6]. Introduce el concepto de *léxico del sistema*. Se trata de un sistema algo limitado a la hora de formular consultas, pues se restringe a preguntas encabezadas por una WH-*word*.[6]
- QuestIO [19,8]. Elimina las limitaciones del anterior y emplea la extracción de datos del contexto para mejorar el proceso de interpretación.
- FREyA [9,8]. Evolución de éste último en el cual se consigue un mayor grado de usabilidad gracias a la implementación de métodos de *feedback*, la interacción con el usuario a través de diálogos de clarificación y el aprendizaje del sistema basado en dichos diálogos.

4. Ask Classora!: Descripción General

4.1. Contexto de Trabajo: de Español a CQL

El desarrollo de interfaces capaces de lidiar con la *ambigüedad* y *expresividad* propias del lenguaje humano [8,3] requiere del empleo de técnicas de Procesamiento

[6] En el caso del inglés, interrogativos como “who”, “where”, “what”, etc.

del Lenguaje Natural (PLN) [13]. Llegados aquí debe tenerse muy presente que, hasta ahora, la mayoría de los trabajos en este ámbito han sido para el inglés, por lo que desarrollar un sistema de este tipo para el español, como es este caso, supone en sí una novedad, a la vez que una dificultad añadida debido fundamentalmente a: *(1)* la mayor complejidad lingüística del español y *(2)* la escasez de recursos y herramientas de PLN libremente disponibles para éste [21].

Por otra parte, Classora emplea como lenguaje formal de consulta el CQL, que actualmente permite formular dos tipos de consultas:

- *Tipo 1.* Devuelve atributos de una unidad de conocimiento. Por ejemplo:
  `fecha de nacimiento, Barack Obama`
  (*"Dime cuándo nació Barack Obama"*)
- *Tipo 2.* Permiten obtener unidades de conocimiento que satisfagan un cierto número de condiciones. Por ejemplo:
  `edificio*, ubicación=Dubai, altura>300 metros`
  (*"Busca los edificios construidos en Dubai que midan más de 300 metros de altura"*)

Nótese que, dadas las actuales limitaciones expresivas de CQL, nuestro prototipo no precisa soportar todas las estructuras sintácticas del idioma de entrada ya que, por ejemplo, ahora mismo no es posible expresar en CQL preguntas encadenadas, coordinaciones disyuntivas o cláusulas relativas, por lo que no es requisito fundamental para nuestro sistema soportar dichas estructuras.

Por otra parte, se ha dotado al sistema de la *flexibilidad* y *extensibilidad* necesarias para que, por diseño, éste no se vea limitado al tratamiento del español como idioma de entrada y del CQL como lenguaje de salida. Asimismo permite también consultas en forma de palabras clave además de como enunciado.

4.2. Proceso General de Interrogación

Tomemos como ejemplo la pregunta *"¿Cuántos años tiene Fernando Alonso?"*. Una vez lanzada la pregunta a través del interfaz web del sistema, ocurre que:

1. La pregunta es enviada al *traductor*, que tratará de interpretarla y traducirla al lenguaje de consulta. Este proceso se describe en la Sección 5.
2. La base de conocimiento es interrogada empleando dicha representación formal de la pregunta original en lenguaje natural.
3. Los resultados iniciales son *enriquecidos* con información extra de interés para el usuario. Tal proceso se describe en la Sección 6.
4. Finalmente, dichos resultados son presentados al usuario.

5. Nuestro *Core*: El sistema de Traducción

El corazón de Ask Classora! es su *traductor*, el cual transforma las consultas en lenguaje natural a su representación en el lenguaje formal de consulta. Para ello nuestro sistema se inspira en conceptos y mecanismos empleados en otros sistemas de propósito similar, si bien integrándolos y extendiéndolos para adaptarlos a su contexto y funcionalidad particulares. Además de los ILN para repositorios de datos estructurados (véase Sección 3.1), queremos destacar los siguientes:

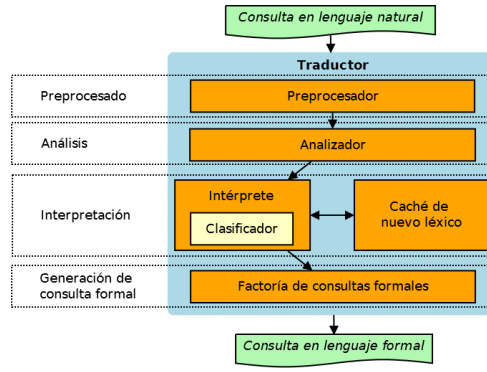

Figura 2. Arquitectura general del traductor.

- De los sistemas *Búsqueda de Respuestas* [17] se han tomado los conceptos de *tipo de pregunta*, *foco de la pregunta*, *tema de la pregunta* y *tipo de respuesta*.
- Las técnicas de *mapeo semántico* [10], que posibilitan la traducción de enunciados en lenguaje natural a su equivalente en lenguaje formal al relacionar términos del primero con conceptos del segundo. Las reglas que implementan dichas relaciones son almacenadas en el denominado *léxico del sistema*.

En base a ello, y tal y como muestra la Figura 2, hemos diseñado un sistema de traducción en cascada donde cada etapa realiza un determinado proceso sobre la consulta, tomando como entrada los resultados obtenidos en la etapa inmediatamente anterior. Describimos a continuación cada una de ellas.

5.1. Preprocesado

Realiza un procesamiento preliminar de la consulta del usuario. El diseño del componente que la implementa permite apilar un número arbitrario de procesos para su aplicación secuencial. Actualmente se restringe a un proceso de corrección ortográfica empleando un API para acceder al corrector de Google.[7]

5.2. Análisis

Permite extraer información de diversa naturaleza (léxica, morfosintáctica, etc.) de la consulta del usuario para su posterior *interpretación*. El *analizador* que la implementa integra tres herramientas de PLN. En primer lugar, el etiquetador morfosintáctico TreeTagger [18] nos permite obtener los *tokens*, *lemas* y *etiquetas morfosintácticas* de la consulta. A su vez, usamos OpenNLP [4] para la detección de *entidades con nombre*. Finalmente, empleamos *gramáticas de estado finito* para la detección de *fechas*. Dichas herramientas han sido previamente entrenadas para el caso del español: para el TreeTagger se han empleado el *Spanish* Crater *corpus* [20] y el lexicón del Callhome *corpus* del LDC [15], mientras que OpenNLP fue entrenado sobre el Conll02 *shared task data* [5].

[7] `https://code.google.com/p/google-api-spelling-java`

5.3. Interpretación

Esta etapa es la encargada de construir los llamados *predicados* (concepto que definiremos luego) apoyándose, entre otros, en un proceso de *mapeo semántico* [10] que relaciona términos de la consulta del usuario con conceptos del léxico del sistema. Esta etapa se implementa a través tres componentes.

Clasificador Su cometido es el de clasificar la consulta del usuario en Tipo 1 o Tipo 2 (véase Sección 4.1) mediante técnicas de Aprendizaje Automático. Emplea para ello el *framework* WEKA [11]. Dado el gran número de algoritmos de clasificación disponibles, nuestra elección se basó en dos premisas: *(1) rapidez de clasificación*, dado que se trata de un sistema interactivo; y *(2)* soporte para *entrenamiento incremental*, ya que facilita la tarea de mejora continua del sistema a partir de la información obtenida de sus *logs* —de los cuales se extraerán nuevas instancias de entrenamiento— sin necesidad de *reentrenar* de nuevo el clasificador desde cero. En base a todo ello, hemos optado por un *clasificador bayesiano* basado en *Naive Bayes* [16, Cap. 7], si bien éste podría sustituirse por cualquier otro clasificador que también cumpla dichos requisitos.

En lo que respecta a la construcción del conjunto de entrenamiento inicial, ésta se llevó a cabo de forma incremental de la siguiente manera:

1. Se incluye una nueva característica (parámetro de clasificación) en el diseño del conjunto de entrenamiento.
2. Se generan una serie de instancias de entrenamiento en base a las características consideradas.
3. Se entrena el clasificador con dicho conjunto de entrenamiento.
4. Se evalúa el clasificador contra las instancias del propio conjunto de entrenamiento. Si el resultado es positivo, se detiene el proceso y se da por válido el conjunto actual; de ser negativo, se introduce una nueva característica en el diseño, siempre y cuando los resultados obtenidos en el incremento actual fuesen mejores que los del anterior —en caso contrario intercambiamos la última característica introducida por la nueva—, para luego volver al Paso 2. Así evitaremos el sobre-entrenamiento (*overfitting*) del clasificador.

En nuestro caso, se obtuvo como resultado el siguiente conjunto de características a emplear en el proceso de clasificación: *(a)* el texto de la consulta, *(b)* etiquetas morfosintácticas de sus términos, *(c)* su longitud, *(d)* número de atributos que aparecen en ella y *(e)* número de entidades que contiene. En lo que respecta al corpus de entrenamiento empleado, éste se describe en la Sección 7.

Intérprete Su función es la de buscar correspondencias entre los términos de la consulta y los conceptos del léxico del sistema, así como la extracción de *predicados* y diversos datos acerca del *contexto de interpretación* de la pregunta. Asimismo, nuestro diseño de este componente le permite implementar diferentes estrategias, cada una de ellas focalizada en el proceso de interpretación de consultas en un idioma concreto, siendo éstas intercambiables dinámicamente.

Entre sus funcionalidades destaca, por su complejidad y relevancia, la *extracción de predicados*. Debe precisarse que, en este contexto, el concepto de *predicado* no es el habitual, sino que corresponde a una estructura (`atributo, operador, valor`) empleada en las consultas Tipo 1 para identificar qué atributo se desea conocer sobre qué entidad, y en las Tipo 2 para establecer las restricciones a cumplir por las entidades resultado. La extracción se realiza leyendo, de izquierda a derecha, los lemas de los términos de la consulta para ir acumulando en el predicado los conceptos resultantes del mapeo semántico.

Lo más difícil de este proceso es la *separación de predicados*, es decir, llegados a cierto punto del proceso determinar si el predicado actual que estamos construyendo está ya completamente formado, para así añadirlo a la lista de predicados extraídos o, por el contrario, si hay que continuar con su construcción. A modo de ejemplo, consideremos que estamos observando el segundo *"en"* de la consulta *"Personas nacidas en España en 1990"*: el valor *"1990"*, ¿pertenece al predicado actual `lugar_de_nacimiento='España'` o forma parte del siguiente?. Para resolver dicho problema optamos por definir un tipo de palabra *delimitador* en base a las etiquetas morfosintácticas obtenidas en el *análisis*, ya que constituyen un buen indicativo de si el lema actual delimita o no un predicado. Sin embargo, tal decisión no se basa únicamente en la aparición de un *delimitador*, pues también se tienen en cuenta otros factores del predicado actual, como si sus campos de `atributo` y `valor` están ya rellenados o si disponemos de un contexto verbal del que sacar partido—concepto que describimos a continuación.

Llegados a este punto conviene recordar que el principal desafío al que hacer frente al procesar el lenguaje humano es su variabilidad inherente [3]: un mismo predicado puede formularse de múltiples maneras, a la vez que una misma expresión, en contextos diferentes, puede dar lugar a diversos predicados. Esto hace que la generación de predicados sea una tarea compleja, si bien el establecimiento de diferentes *contextos de interpretación* puede sernos de gran ayuda:

- *Contexto de la pregunta*. Se corresponde con la unidad de conocimiento principal de la que se desean datos. Esto permite dirigir la búsqueda durante el mapeo entre términos de la consulta y conceptos del léxico. Para identificarlo observamos cuál es la unidad de conocimiento que más veces aparece relacionada con los posibles conceptos correspondientes a los términos de la consulta. Por ejemplo, en *"¿Cuántos años tiene Fernando Alonso?"*, la unidad de conocimiento principal es `persona` (contexto de la pregunta), lo que permite identificar que *"años"* se refiere al atributo `edad-antiguedad` de `persona`, y no al de otras entidades que lo comparten (p.ej. `organizacion`).
- *Contexto verbal*. Los conceptos asociados a un término previamente etiquetado como verbo, facilitan no sólo el poder rellenar los campos correspondientes del predicado actual, sino también los de posibles predicados subsiguientes. Por ejemplo, para *"Personas nacidas en España en 1990"*, es preciso guardar los conceptos asociados a *"nacer"*: `fecha_de_nacimiento` y `lugar_de_nacimiento`; pues tras generar (`lugar_de_nacimiento`, `=`, `España`), aún restará por generar el predicado (`fecha_de_nacimiento`, `=`, `1990`), que hará uso de ese segundo concepto asociado a *"nacer"*.

Asimismo, al buscar correspondencias entre los términos de la pregunta y el léxico del sistema, empleamos los lemas de las palabras, lo que permite abordar el problema de la variación flexiva (variaciones de género, número, tiempo, etc.), reduciendo así el tamaño del léxico y haciendo el componente más robusto.

Puede también ocurrir que, dado el contexto actual, haya más de una correspondencia posible entre un término de la consulta y los conceptos del léxico. En ese caso, el sistema solicitará ayuda al usuario a través de un *diálogo de desambiguación de términos*, para que sea él quien determine cuál de esos conceptos, mostrados como una lista de opciones, es el adecuado. Por ejemplo, de nuevo en el contexto de la unidad de conocimiento `persona`, el término *"localidad"* podría estar relacionado tanto con `lugar_de_nacimiento` como con `lugar_de_defuncion`. Tal situación podría ser resuelta por el propio sistema si dispusiese de más información sobre el contexto de interpretación de la consulta, y es dicha información la que se pretende obtener mediante esta clase de diálogo.

Caché de Nuevo Léxico Este componente dota al sistema de su capacidad de aprendizaje en base a los datos obtenidos con los *diálogos de clarificación*, diálogos interactivos en los que, tras toparse en la consulta con un término desconocido sin correspondencia en el léxico del sistema, éste pide ayuda al usuario para hacer el mapeo. De nuevo en el contexto de `persona`, un ejemplo sería que el sistema no fuese capaz de relacionar el término *"origen"* con el concepto `lugar_de_nacimiento`, debiendo ser el usuario quien, mediante un diálogo de clarificación, indicase tal relación. El sistema mantendrá, dentro de la *caché*, una lista con todos los posibles nuevos conceptos (*candidatos*) relacionados con atributos, unidades de conocimiento u otros conceptos obtenidos del contexto en el que apareció el término desconocido. Cada uno tiene asociada una *puntuación* que se irá incrementando cada vez que un usuario nombre a dicho candidato dentro de un diálogo de clarificación. Cuando esta puntuación sobrepase cierto umbral, se procederá a introducir el nuevo concepto en el léxico del sistema.

Como se puede ver, ambos tipos de diálogo dotan al sistema de una mayor autonomía de cara a futuras situaciones similares. Sin embargo, debemos recordarle al lector que los mecanismos de diálogo son el "último recurso", ya que únicamente se aplican en caso de que los métodos de desambiguación y mapeo automáticos del sistema no hayan producido resultados satisfactorios.

5.4. Generación de la Consulta en Lenguaje Formal

Esta última etapa es la que encargada de componer la consulta CQL final. El diseño empleado le permite disponer de varias estrategias de generación, dotando así al sistema de mayor flexibilidad y facilidad de administración, además de permitir la posibilidad de trabajar con diversos lenguajes formales de consulta.

En el caso concreto de CQL, tendremos que prestar atención al tipo de consulta (ver Sección 4.1) detectado, según el cual realizaremos el mapeado entre los componentes de los predicados obtenidos en la etapa de interpretación y los de la consulta CQL en construcción.

6. Enriquecimiento de Resultados

Tras interrogar a la base de conocimiento con la consulta generada por el traductor, Ask Classora! muestra, junto con los resultados devueltos, un breve desglose del proceso para su obtención, proveyendo al usuario de una justificación de la salida obtenida, así como de cierto entendimiento de la lógica seguida durante la interpretación de su consulta. Esto le ayudará a formular futuras consultas de forma más efectiva. Se presentan también *consultas relacionadas* formuladas anteriormente por otros usuarios y almacenadas en un *log*.

Como ejemplo, para *"¿Qué ocupación tiene Fernando Alonso?"* se indicaría que el sistema ha interpretado que la pregunta (clasificada como Tipo 1) se refiere a una `persona` (*unidad de conocimiento*) y que se desea conocer la `profesión` (*atributo*) de la entidad *"Fernando Alonso"*, mostrando además las consultas relacionadas *"trabajo fernando alonso"* y *"¿en qué trabaja Fernando Alonso?"*.

7. Evaluación del Sistema de Traducción

Nuestras pruebas iniciales se centraron en la unidad de conocimiento `persona`, empleando un conjunto de consultas creado por personal externo y con plena libertad sobre cómo formularlas. El conjunto resultante[8] está formado por 202 consultas: 60 de Tipo 1, abarcando los diferentes atributos posibles sobre los que preguntar; y otras 142 de Tipo 2, cubriendo las diferentes combinaciones posibles de atributos (entre uno y cinco) en cuanto a las restricciones que se pueden imponer en la consulta. Por otra parte, adaptando a nuestro contexto la propuesta original de [10], hemos definido las siguientes métricas de evaluación:

- *Porcentaje de traducciones correctas.* Busca cuantificar el número de resultados correctos desde la perspectiva del usuario, analizando para ello cuántas de las traducciones devueltas dieron lugar a la respuesta deseada.
- *Porcentaje de predicados correctos.* Se refiere al número de veces en las que la extracción de predicados de la consulta se hizo adecuadamente, independientemente de si la posterior traducción a lenguaje formal fue correcta.

De lanzarse un diálogo de desambiguación, dicha interpretación se considerará correcta si en la lista de opciones aparece el concepto adecuado al término ambiguo. Asimismo, un diálogo de clarificación contará siempre como fallo.

El Cuadro 1 muestra los resultados obtenidos en nuestras pruebas, a la vista de los cuales podemos extraer varias conclusiones:

1. La adecuación de nuestro planteamiento se ve apoyada por el alto porcentaje de aciertos obtenido, especialmente en el caso de consultas Tipo 1, más sencillas y menos sensibles a la variabilidad del lenguaje.
2. Se detectaron varios casos en los que la clasificación de la pregunta fue incorrecta. El tipo de la pregunta juega un papel esencial en la elección de la

[8] Disponible en: `https://www.dropbox.com/sh/r9zw5qi7cky8irb/Zn8F3sDZjp`

Cuadro 1. Resultados de evaluación.

	Traducciones correctas	Predicados correctos
Tipo 1	100 %	100 %
Tipo 2	87,94 %	95,07 %
Total	**91,58 %**	**97,02 %**

estrategia de generación de la consulta formal, por lo que un error de clasificación redunda en una traducción incorrecta a partir de unos predicados correctos. Esto podría evitarse en un futuro empleando un conjunto de entrenamiento del clasificador más amplio, el cual, con los recursos disponibles en la actual fase de demostración, no ha sido posible obtener.

3. Finalmente, un análisis en profundidad de los errores cometidos en la formación de predicados, reveló que, con frecuencia, las causantes habían sido las herramientas de terceros integradas en el sistema. Así, tanto los fallos durante la corrección ortográfica, como aquéllos en el etiquetado morfosintáctico, provocaron que, en la mayor parte de estos casos, el intérprete fuera incapaz de generar predicados correctos.

8. Conclusiones y Trabajo Futuro

Hemos presentado en este trabajo un demostrador tecnológico para estudiar la viabilidad de la posible integración de una interfaz de lenguaje natural en la base de conocimiento comercial *online Classora Knowledge Base*. Nuestras pruebas han mostrado un rendimiento claramente positivo, además de un comportamiento robusto. Se trata, además, de un diseño flexible y fácilmente extensible.

La existencia de una demanda de mercado al efecto en el caso del español, unida a que el inglés ha venido siendo la lengua dominante en este campo, hacen de nuestro sistema una propuesta de interés a la vez que novedosa en ese aspecto.

Respecta al futuro, debemos considerar la utilización de los *logs* del sistema para, mediante técnicas de aprendizaje automático, ampliar y mejorar de forma continua tanto el léxico del sistema como el afinamiento del clasificador. En cuanto al traductor, pretendemos dar cabida, en cuanto CQL lo permita, a cláusulas y pronombres relativos, así como a preguntas encadenadas. Finalmente, queremos extender Ask Classora! al caso del inglés.

Como apunte final, el sistema está listo para su pronto despliegue en un entorno de producción real, si bien todavía en fase de demostración.

Agradecimientos: Trabajo parcialmente financiado por el Ministerio de Economía y Competitividad y FEDER (TIN2010-18552-C03-02) y por la Xunta de Galicia (CN2012/008, CN2012/319).

Referencias

1. I. Androutsopoulos, G. D. Ritchie, and P. Thanisch. Natural Language Interfaces to Databases – An Introduction. *Natural Language Engineering*, 1(1):29–81, 1995.
2. ASF. Apache OpenNLP models. `http://opennlp.sourceforge.net/models-1.5/`
3. A. Arampatzis, T. P. van der Weide, Patric van Bommel, and C. H. A. Koster. Linguistically-Motivated Information Retrieval. In *Encyclopedia of Library and Information Science*, vol. 69, pp. 201–222. Marcel Dekker, Inc., 2000.
4. ASF. The Apache OpenNLP Project. `http://opennlp.apache.org`
5. ACL. CONLL02 shared task data. `http://www.clips.ua.ac.be/conll2002/ner/`
6. P. Cimiano, P. Haase, J. Heizmann, M. Mantel, and R. Studer. Towards portable natural language interfaces to knowledge bases – The case of the ORAKEL system. *Data & Knowledge Engineering*, 65(2):325–354, 2008.
7. E. F. Codd. Seven Steps to Rendezvous with the Casual User. In *Proc. of the IFIP Working Conference on Data Base Management*, pp. 179–200, 1974.
8. D. Damljanović. *Natural Language Interfaces to Conceptual Models.* PhD thesis, The University of Sheffield, Department of Computer Science, 2011.
9. D. Damljanović, M. Agatonovic, and H. Cunningham. FREyA: An interactive way of querying linked data using natural language. In *The Semantic Web: ESWC 2011 Workshops*, vol. 7117 of *LNCS*, pp 125–138. Springer-Verlag, 2012.
10. M. Gao, J. Liu, N. Zhong, F. Chen, and C. Liu. Semantic mapping from natural language to OWL queries. *Computational Intelligence*, 27(2):280–314, 2011.
11. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.
12. M. A. Hearst. *Search User Interfaces.* Cambridge University Press, 2009.
13. D. Jurafsky and J. H. Martin. *Speech and Language Processing (2nd ed.)* Pearson–Prentice Hall, 2009.
14. E. Kaufmann and A. Bernstein. How Useful are Natural Language Interfaces to the Semantic Web for Casual End-Users? In *The Semantic Web*, vol. 4825 of *LNCS*, pp. 281–294. Springer-Verlag, 2007.
15. LDC. The CALLHOME corpus. `https://www.ldc.upenn.edu`
16. C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* The MIT Press, 1999.
17. J. Prager. Open-Domain Question Answering. *Foundations and Trends in Information Retrieval*, 1(2):91–231, 2006.
18. H. Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proc. of the Int. Conf. on New Methods in Language Processing*, pp. 44–49, 1994.
19. V. Tablan, D. Damljanović, and K. Bontcheva. A natural language query interface to structured information. In *Proc. of the 5th European Semantic Web Conference (ESWC'08)*, vol. 5021 of *LNCS*, pp. 361–375. Springer-Verlag, 2008.
20. Lancaster University. Spanish CRATER corpus. `http://www.ling.lancs.ac.uk/`
21. J. Vilares, M. A. Alonso, and M. Vilares. Extraction of complex index terms in non-English IR: A shallow parsing based approach. *Information Processing & Management*, 44(4):1517–1537, 2008.
22. D. H. Warren and F. C. Pereira. An efficient easily adaptable system for interpreting natural language queries. *Computational Linguistics*, 8(3-4):110–122, 1982.
23. W. A. Woods, R. M. Kaplan, and B. Nash-Webber. The Lunar Sciences Natural Language Information System: Final Report. BBN Report 2378, Bolt Beranek and Newman Inc., 1972.