Question answering de dominio abierto y de dominio cerrado

Julián Peller

Abril 2016

index.tex

index.tex

index.tex

index.tex

index.tex

index.tex

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

index.tex

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

index.tex

index.tex

Introducción

- Sistema que implementa el modelo propuesto por Popescu en [Popescu et al. 2003a] y [Popescu et al. 2003b].
- Define concepto de tratabilidad semántica y traduce preguntas semánticamente tratables en consultas SQL
- Código implementado en java, accesible en http://github.com/julian3833/popescu-world
- Testeado sobre base de datos relacional de juguete provista por MySQL, llamada World, con información geográfica básica de países, ciudades e idiomas.
- Sistema de funcionalidad acotada con soporte solo para el inglés.

index.tex

Dominio de problemas

Dominio acotado y específico

- Restaurants, bancos, vuelos, libros, etcétera (pero solo uno)
- Bases de datos relacionales $\rightarrow$ datos estructurados
- Question answering como interfaz para una base de datos
- Traducir pregunta a SQL

Ejemplo

Pregunta

When did Albert Einstein die?

Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'

Respuesta

April 18th, 1955

Ejemplo

Pregunta

When did Albert Einstein die?

Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'

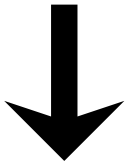
Respuesta

April 18th, 1955

Ejemplo

Pregunta

When did Albert Einstein die?



Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'

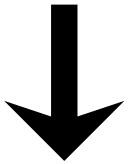
Respuesta

April 18th, 1955

Ejemplo

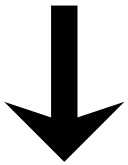
Pregunta

When did Albert Einstein die?



Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'



Respuesta

April 18th, 1955

Modelo teórico

- Popescu et al. $\rightarrow$ QADB, Precise
- Tratabilidad semántica de una pregunta q en el contexto de una DB d.
- Las preguntas no tratables se rechazan, las tratables se traducen.
- Motivación: no se puede fallar activamente (mal interpretar)

Tratabilidad semántica / motivación

- La complejidad de las preguntas en lenguaje natural es arbitrariamente grande.
- Las preguntas semánticamente tratables son preguntas fáciles pero abarcativas
- Distinguir un subconjunto 1) *tratable* y 2) *abarcador*
- Rechazar y pedir reformulación de las no tratables
  - Es mejor no dar respuesta a dar una mala. Conservar la confianza en el sistema.

Tratabilidad semántica / idea coloquial

Tratabilidad semántica en el contexto de una DB

- Una Q-word (Qué, quién, cuándo, dónde, etc.)
- Pares de atributos y valores
- Valores sueltos
- Términos no significativos y menciones a relaciones
- Por ejemplo:
  - ¿Qué bancos de la empresa Credicoop están localizados en el barrio de Coghlan?
  - ¿Quién era el presidente de México en 1993?

Definiciones (1): Elemento, qword, compatibilidad, token, marcador sintáctico

- Elementos de una DB: Relaciones, Atributos y Valores
- Qwords - pronombres interrogativos
  - {What, where, which, when, who}
  - {Qué, dónde, cuál, cuándo, quién}
- Compatibilidad
  - Valor con sus atributo
  - Atributos con sus relaciones
  - Valor con las relaciones de sus atributos
  - Q-words compatibles con atributoes
    - Definición a mano especifica por DB
- Token: un conjunto de lemas de palabras de la pregunta que corresponden a un elemento de la base de datos.
  - Por ejemplo, {experiencia, requerir} y {experiencia, necesario} son dos tokens para "Experiencia Requerida"
- Marcador sintáctico: palabras que no aportan a la interpretación de la pregunta, definidas a mano

Definiciones (2): Correspondencia entre tokens y elementos

- Cada elemento de la base de datos se separa en palabras individuales:
  - "Experiencia Requerida" $\rightarrow$ {Experiencia, Requerida}
- Se genera un conjunto de sinónimos para cada palabra usando Wordnet:
  - experiencia $\rightarrow$ {experiencia, conocimiento, habilidad,...}
  - requerida $\rightarrow$ {requerida, necesaria, indispensable,...}
- Se toman los lemas o raíces de todas las palabras
  - experiencia $\rightarrow$ {experiencia, conocimiento, habilidad,...}
  - requerida $\rightarrow$ {requerir, necesario, indispensable,...}
- Se generan tokens combinando los lemas de los sinónimos de cada elemento:
  - tokens = {(experiencia, requerir), (conocimiento, requerir), (habilidad, requerir),(experiencia, necesario), (conocimiento, necesario), (habilidad, necesario),(experiencia, indispensable), (conocimiento, indispensable), (habilidad, indispensable)}
- Este conjunto de tokens son los tokens que se corresponden con el elemento
- Un token tiene tipos dependiendo a qué elementos corresponda

Definiciones (3): tokenización completa, asociación sintáctica

- **Tokenización completa de una pregunta *q***: cualquier conjunto de tokens en los que cada término que no es un marcador sintáctico de q aparece en exactamente un token del conjunto.
  - Una partición de tokens (como fueron definidos) de la pregunta q
- **Asociación sintáctica**: dos tokens están sintácticamente asociados en q si cumplen ciertas restricciones sintácticas.
  - Modelado con una función *attachment*(t1, t2, q) $\rightarrow$ *booleano*

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (5): Semánticamente tratable

Si una pregunta tiene solo una q-word y al menos una traducción válida entonces es **semánticamente tratable**

Y cada traducción válida de q determina una posible consulta SQL:

SELECT	Elementos apareados con qwords
WHERE	Pares de atributos y valores generados por el Matcher
FROM	Todas las relaciones mencionadas

Precise

El sistema utilizado por Popescu et al.

- Dada una pregunta q, determina si es semánticamente tratable y si lo es, genera una consulta SQL correspondiente. Si no, la rechaza pidiendo una reformulación.

Lexicon

- Sinónimos lematizados de todas las relaciones, atributos y valores
  - Wordnet
- Determina qué términos son comprensibles de una pregunta
- Las preguntas tratables tienen solamente marcadores sintácticos + palabras del lexicon.

Matcher

- La pieza clave.
- Relaciona cada valor con un único atributo, implícito o explícito usando un algoritmo de max flow.

Precise / módulos

- Lexicon: encargado de generar, para cada elemento de la base de datos, el conjunto de tokens sinónimos que se utilizará para verificar correspondencia.
- Tokenizer: encargado de generar todas las tokenizaciones completas de la pregunta y, para cada token, consultar al lexicon y retornar la lista de elementos de la base de datos que le corresponden.
- Matcher: encargado de verificar que las tokenizaciones completas y los elementos correspondientes generados por el tokenizer cumplan las condiciones 1 a 3 utilizando el modelo de grafos y el algoritmo max-flow recién ilustrado.
- Parser plug-in: el módulo computa la función de asociación sintáctica, basándose en el parse tree de Charniak.
- Query generator: dado el conjunto de elementos de la base de datos traducido de la pregunta, genera una query SQL.
- Equivalence Checker: verifica si diferentes queries son la misma formulada de diferentes maneras.

Ejemplo

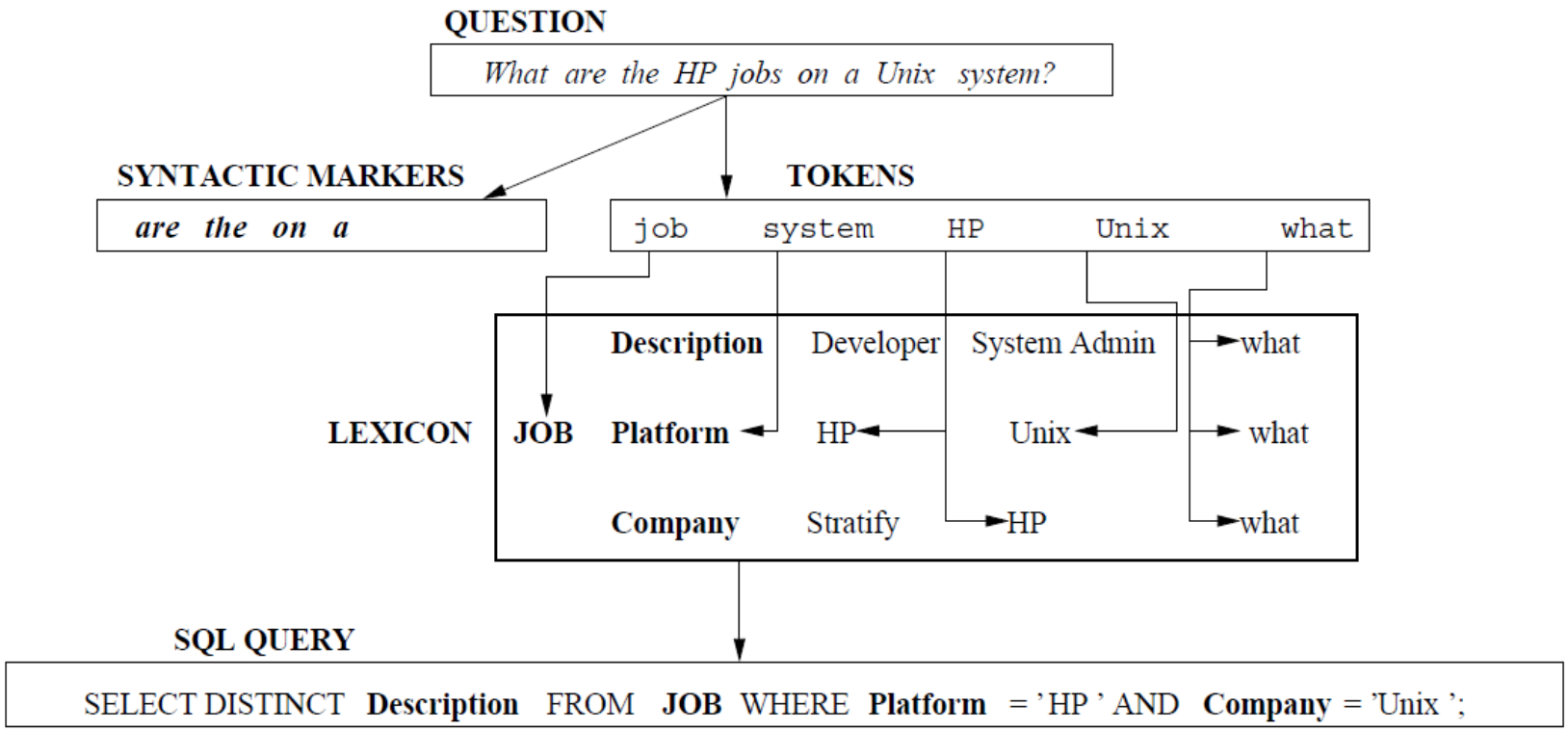


Figura: La traducción de la pregunta "What are the HP jobs on a Unix system?" a una consulta SQL

Ejemplo

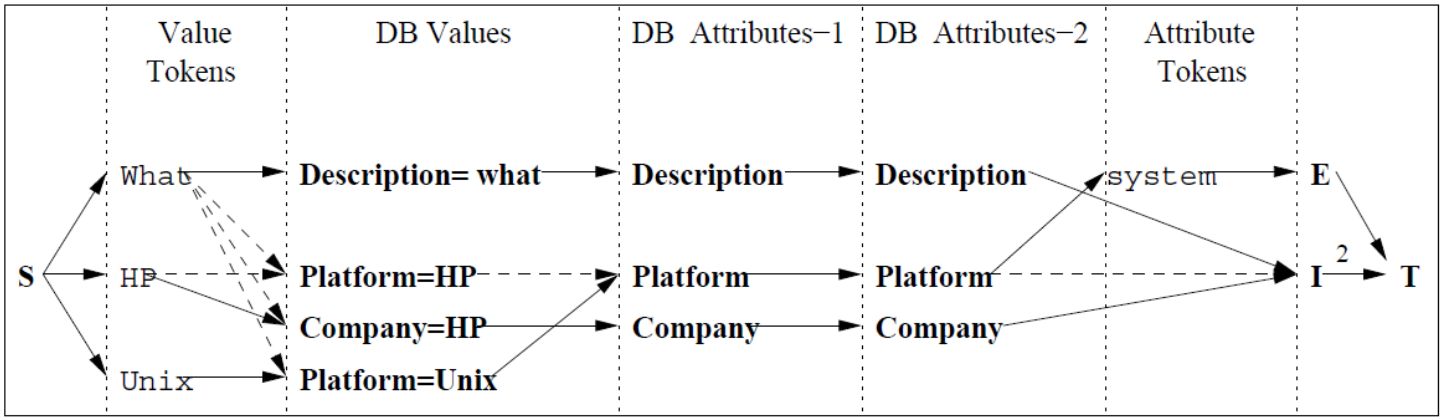
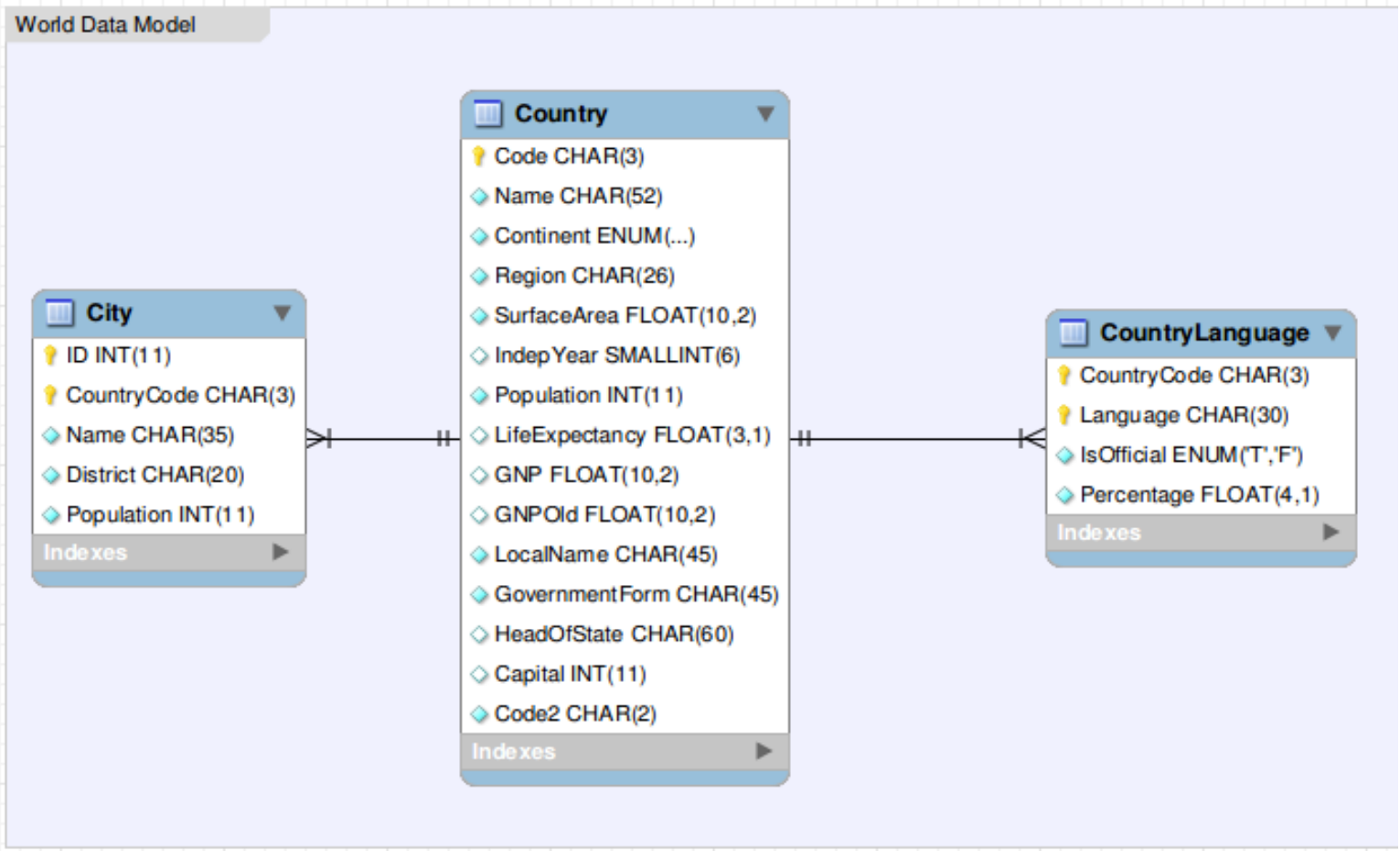


Figura: El grafo de atributos y valores creado por Precise para la pregunta “What are the HP jobs on a Unix system?”

index.tex

Base de datos: World (Country, City y CountryLanguage)



Lexicón

El lexicón es el módulo encargado de generar un conjunto de tokens para cada elemento de la base de datos. Una vez construido este conjunto, las responsabilidades del módulo son las siguientes:

- Dado un lema, devolver el conjunto de tokens que lo contienen (*getTokens*()).
- Dado un token, devolver el conjunto de elementos de la base de datos que le *corresponden* (*getMatchingElements*()).
- Wordnet. TokenAugmenter. Polisemia.

TokenAugmenter

Elemento original	Sinónimos
head of state	president, leader, emperor, king
region	location
surface area	size, total size, square kilometers, km2
independence year	independent, independency

Cuadro: Sinónimos introducidos por el Token Augmenter

Compatibilidad de Qwords

Qword	Atributos relacionados
What	**Name**, District, Population, Code, Continent, SurfaceArea, LifeExpectancy, GNP, LocalName, GovernmentForm, Capital, IsOfficial, Percentage, Region
Which	Los mismos que para what
Where	**Region**, Continent, Capital, District
Who	**HeadOfState**
When	**IndependenceYear**

Cuadro: Atributos compatibles con cada Qword

Tokenizer

1. Separar la pregunta en palabras, eliminar puntuaciones y pasar a lower case.
2. Lematizar las palabras. Para esto usamos Freeling
3. Eliminar marcadores sintácticos.
4. Para cada lema, obtener todos los tokens que lo contienen del Lexicon (método getTokens).
5. Para cada token potencial (resultado del paso anterior) verificar que todos sus lemas estén presentes en el conjunto de lemas de la pregunta original.
6. Generar el conjunto de partes de todos los tokens hasta aquí obtenidos. Probando con cada elemento del conjunto de partes en lugar de utilizar solamente el conjunto original podemos obtener subconjuntos que cumplan también la condiciones requeridas para considerarse una tokenización completa (evaluados en 7).
7. Para cada uno de estos subconjuntos, verificar 1) que sus tokens cubran completamente los lemas significativos de la pregunta original y 2) que no haya lemas repetidos entre los tokens.

Matcher

- Input = tokenizaciones completas generadas por el Tokenizer,
- Construye el grafo que expusimos en la sección NADA
- Algoritmo de max flow. [1][2]
- Las aristas implicadas en el flujo máximo posible asocian
  - 1) los tokens de valor y de atributo y los correspondientes elementos (valores y atributos, respectivamente) de la base de datos y
  - 2) pares de valores y atributos entre sí.
- Otras soluciones posibles. Si el flujo es igual a la cantidad de tokens de valor en la pregunta es potencialmente una traducción válida. Retiramos ordenadamente las aristas del grafo que ocurren entre la columna 2 y 3 (tokens de valor y valores de la db).

[1] El problema de max-flow es un problema de grafos que consiste en "enviar" el máximo flujo posible a través de un grafo dirigido con dos nodos especiales (fuente o source y sumidero o sink) y aristas con cierta capacidad mayor o igual que cero. Este flujo debe ir desde el nodo fuente al nodo sumidero, respetando las capacidades de las aristas y respetando que, para cada nodo, el flujo saliente no puede ser mayor al flujo entrante.

`http://web.mit.edu/~ecprice/acm/acm08/MaxFlow.java`

Matcher: condiciones

Finalmente, verificamos cuales de las soluciones con máximo flujo cumplen las condiciones requeridas para ser una traducción válida según enunciamos en **??**:

1. Todos los tokens de la tokenización tienen un único elemento de la base de datos asociado y no hay elementos de la base de datos repetidos. (Mapping.meetsConditionOne())
2. Cada token de atributo se relaciona con un único token de valor respetando que: (Mapping.meetsConditionTwo())
   1. el atributo relacionado con el token de atributo y el valor relacionado con el token de valor son compatibles (esta condición está garantizada por el max-flow mismo)
   2. ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones: (Mapping.meetsConditionThree())
   1. la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   2. ambos tokens (token de relación - token de atributo o bien

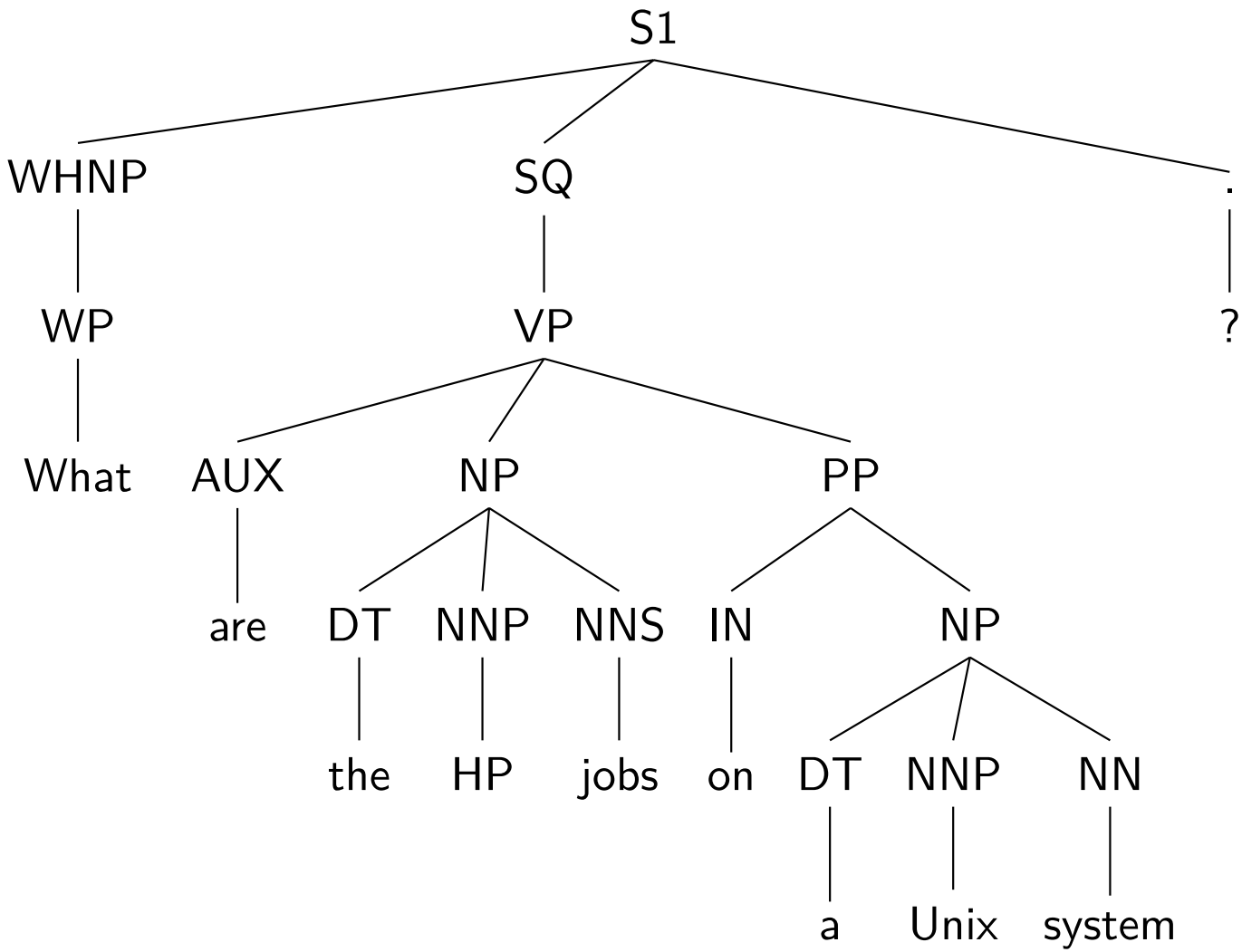
Matcher: notas

La definición de las reglas se hizo a prueba y error y es posible que sea simple, para alguien con mayores conocimientos lingüísticos, mejorarlas. Probablemente tenga sentido diferenciar entre sustantivos principal y modificador (Unix y system, por ejemplo) y modificar las reglas 3 y 4 para conectar solo los principales, entre otras mejoras. El sistema ofrece también la opción de no evaluar las asociaciones sintácticas en absoluto, en cuyo caso se mejora la performance pero aparecen nuevas traducciones válidas que podrían haber sido filtradas por estas condiciones, que el usuario deberá desambiguar manualmente). Además, estas reglas seguramente sean insuficientes y estén dejando afuera varias preguntas que deberían poder procesarse.
Finalmente, todos los resultados de max-flow que cumplen con las condiciones de 1 a 4 son traducciones válidas, que pasan al MappingFilter, que realiza ciertos filtrados que describiremos, de nuevo, en un título aparte. El resultado de este módulo es una lista de Mappings (una estructura que contiene 1) una tokenización completa de la pregunta original y 2) un mapeo válido entre cada token de la misma en un elemento de la base de datos). Cada mapeo es una traducción válida de la pregunta. Si existe solo uno, entonces este mapeo se traducirá en una query que generará el resultado. Si no, corresponde al MappingFilter realizar filtrados inteligentes de las múltiples soluciones y, en caso de que continuen existiendo múltiples soluciones, entonces se consultará al usuario qué quiso preguntar. Por otro lado, si no fue posible generar ninguna traducción válida, se retornará al usuario sin

Charniak

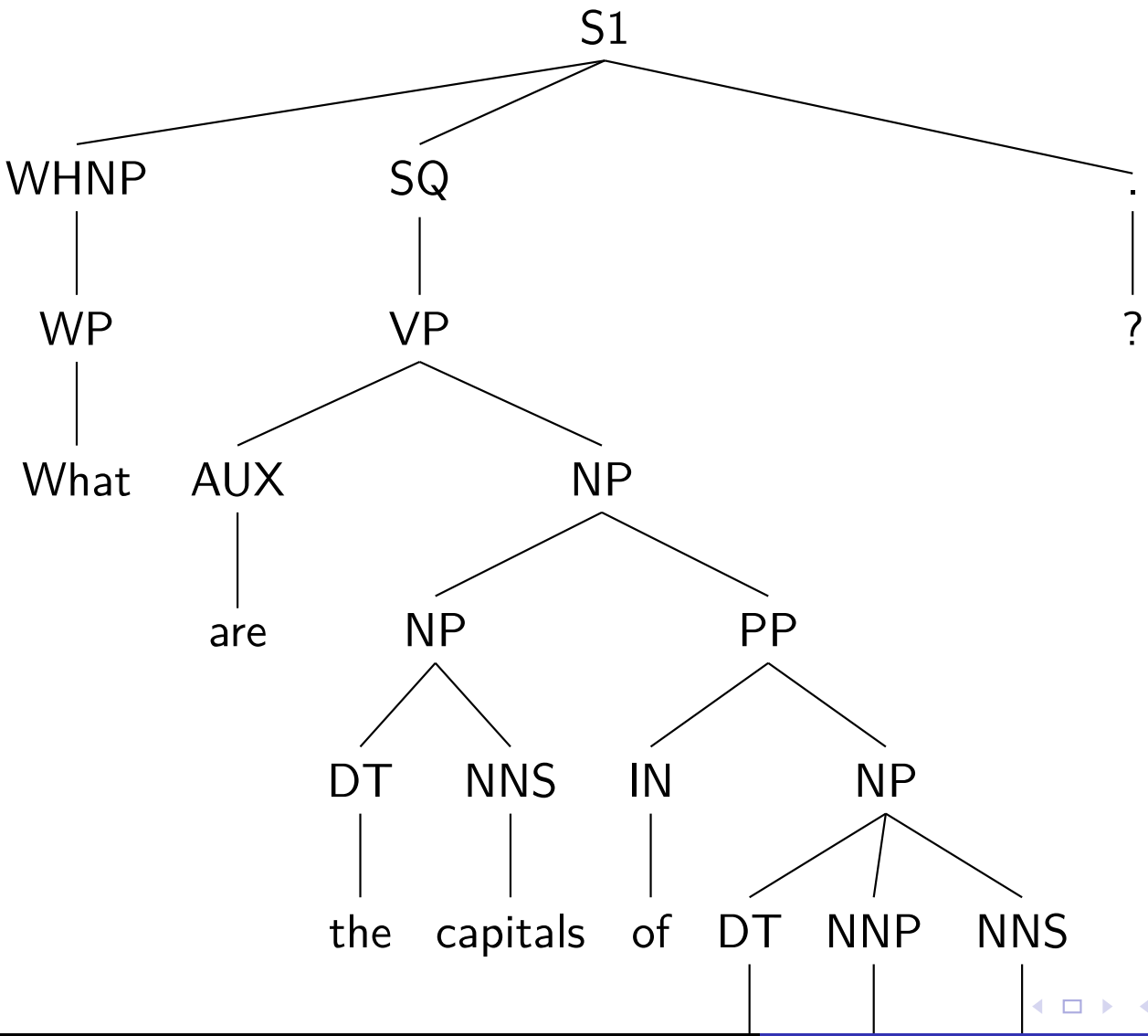
Para verificar las condiciones de asociación sintáctica (2.b y 3.b) utilizamos la implementación oficial del árbol sintáctico de Charniak (`github.com/BLLIP/bllip-parser`), con un wrapper en java que es la clase *uba.nlp.CharniakParseTree*. Las reglas sintácticas utilizadas por Precise no están especificadas en los trabajos, por lo que creamos nuestras propias reglas tomando los ejemplos de asociaciones sintácticas dadas en los trabajos, evaluando los resultados de diferentes ejemplos y extrapolando reglas a partir de allí.
Veamos, en primer lugar, ejemplos del parse tree para tres preguntas:

Charniak



"What are the HP jobs on a Unix system?"

Charniak


S1
WHNP
SQ
.
WP
VP
?
What
AUX
NP
are
NP
PP
DT
NNS
IN
NP
the
capitals
of
DT
NNP
NNS

Charniak: reglas

as reglas que construimos son las siguientes (notar que los nombres pueden denotar una direccionalidad, pero las relaciones son simétricas). Utilizamos NNx para denotar cualquier hoja sustantivo, adjetivo o adverbio (JJ, JJR, JJS, NN, NNS, NNP, NNPS, RB, RBR, RBS, CD), NN para denotar estrictamente sustantivos (NN, NNS, NNP, NNPS), Nx para denotar cualquier sintagma relacionado a estos tipos de hojas (nodos intermedios): (ADJP, ADVP, NP, S) y Wx para denotar alguna de estas hojas: WDT, WP, WP$ y WRB.

1. Hermanos:
   - NNx $\Leftrightarrow$ NNx $\Leftrightarrow$ Wx si comparten el mismo el mismo padre
2. Qwords a sustantivo:
   - Una Wx dentro de un WHNP $\Rightarrow$ un NN dentro de uno o más Nx dentro de un VP dentro de un SQ con el mismo SBARQ como padre que la WHNP mencionada.
3. Sintagma nominal a sintagma preposicional:
   - NN dentro de un Nx (1) $\Rightarrow$ NNx dentro de uno o más Nx dentro de un PP con el mismo padre que el Nx marcado con el (1)

MappingFilter, QueryGenerator, MainProcessor

MappingFilter

La clase *uba.app.MappingFilter* es responsable, en primer lugar, de quitar las traducciones válidas repetidas y, en segundo lugar, de aplicar una serie de reglas también de filtrado. Para eliminar las traducciones repetidas, las transformamos en queries y simplemente comparamos por igualdad, ya que el generador de queries ordena las cláusulas generando, para inputs iguales pero diferentemente ordenados, la misma query. Además de este filtro, aplicamos tres reglas más, de creación propia. En primer lugar, eliminamos "semi duplicados", que consisten en atributos similares en la base de datos, en concreto, para esta base de datos, consideramos semi-duplicada a una query que consulta por Country.Name y por Country.LocalName solamente. La segunda y la tercera regla están basadas en la siguiente idea: en el caso en que exista más de una traducción válida en la cual la qword esté asociada con un atributo *implícito*, por como está el sistema definido, también estará asociado con todos los atributos que pueda estarlo (ver tabla table:atributos-qwords para las compatibilidades definidas). Esto implica que, por ejemplo, para cualquier pregunta cuya qword sea 'what' y no tenga su atributo asociado explícito, habrá 16 diferentes traducciones válidas. En los trabajos sobre los que basamos nuestro proyecto este problema no está mencionado y en los ejemplos presentados no es visible. Lo que hicimos en este caso es aplicar dos conceptos de preferencia. La segunda regla prefiere, entre los atributos implícitos posibles, aquellos cuya relación

Ejemplos

- FILL ME
- Who is the head of state of Zimbabwe?
- What caribbean countries are also considered north american?

Conclusiones y limitaciones

- FILL ME

index.tex

index.tex

Introducción

- Sistema de dominio abierto multilingüe
- Basado en Qanus, adaptado para usar Freeling.
- Ejercicios de CLEF'07 monolingüe para español y portugués
- Wikipedia como base de conocimiento

Adoptamos el framework Qanus y el sistema QA-sys para:

1. Incorporar Freeling como librería de herramientas de procesamiento del lenguaje permitiendo un funcionamiento multilenguaje.
2. Trabajar sobre el corpus de datos y preguntas de los ejercicios de CLEF '07 de español y portugués
3. Incorporamos y evaluamos algunas mejoras tomadas de la reseña a la literatura científica

index.tex

Dominio de problemas y enfoque

Dominio general y universal

- Cualquier cosa
- Corpora de texto $\rightarrow$ datos no estructurados
- Question answering como sistema de pipeline
- IR + NLP + Heurísticas

Marco teórico

Consenso general sobre la arquitectura: pipeline de al menos 3 componentes:

- Módulo de procesamiento de la pregunta
  - Submódulo principal: Question Classifier
  - Anotaciones de la pregunta (NER, POS, QC, Otros: focus, answer type)
  - Expansión/reformulación de la consulta (Heurística de Lampert, Moldovan)
- Módulo de procesamiento de documentos
  - Submódulo principal: índice invertido (estructuras de IR)
  - (Otros: división en parágrafos, filtrado grueso, re-ranking)
- Módulo de procesamiento de la respuesta
  - Valor agregado ppal de un sistema de QA sobre uno de IR
  - El menos estandarizado de los tres
  - Heurísticas puntuales basadas en casos.

Heurística de generación de queries [Lampert, 2004]

8 pasos

1. Palabras no stop words entre comillas
2. Entidades nombradas reconocidas
3. Construcciones nominales con sus adjetivos
4. Demás construcciones nominales
5. Sustantivos con sus adjetivos
6. Demás sustantivos
7. Verbos
8. El 'focus' de la pregunta

Competencias y métricas

- FILL ME

index.tex

Ejercicios

Cross-Language Evaluation Forum

- Principal organizador de conferencias de question answering multilingüe para idiomas europeos.
- La tarea principal de '07 presentaba una complejidad adecuada para el scope de esta tesis.

Ejercicios

- Subset de dos ejercicios de la tarea principal de question answering de '07.
- Corpora, inputs de prueba y resultados esperados para una experimentación sobre diferentes idiomas disponible.
- Wikipedia + newsletters como corpus
  - Nosotros trabajamos solo con las wikipedias
- 200 preguntas (factoids, definition o list)
- Respuesta exacta. Topics + clusters + co-referencia.
- Preguntas sin respuesta
- Tomamos ejercicios monoligüe es-es y pt-pt (español-español y portugués-portugués)

Tareas activadas en CLEF '07

Table 1: Tasks activated in 2007 (in green)

SOURCE LANGUAGES (questions) \ TARGET LANGUAGES (corpus and answers)	BG	DE	EN	ES	FR	IT	NL	PT	RO
BG	■		■	■					
DE		■	■	■					
EN	■	■		■	■	■	■	■	■
ES			■	■	■			■	
FR			■	■	■				
IN			■						
IT		■	■	■		■			
NL			■	■			■		
PT		■	■	■	■			■	
RO			■	■					■

Figura: Tareas activadas en la competencia Clef '07

Preguntas

Subset	Idioma	Factoid	Definition	List	Total
Todo	es	158	32	10	200
	pt	159	31	10	200
Wiki	es	130	24	9	163
	pt	104	18	8	130

Cuadro: Totales por tipo de pregunta

Ejemplo de grupo con tema y correferencia (primer cluster de preguntas para es-es):

- ¿En qué colegio estudia Harry Potter?
- ¿Cuál es el lema del colegio?
- ¿En qué casas está dividido?
- ¿Quién es el director del colegio?

Qanus

Question-Answering @ National University of Singapore

- Un framework de question answering basado en information retrieval
- QA-sys: un sistema de QA funcional simple construido sobre este framework.
- Actualizado por última vez en noviembre de 2012, código abierto
- Herramientas de NLP de Stanford (POS, NER y QC, para inglés) y Lucene.
- Estructura de Pipeline. Orientado para TREC 07 (Aquaint)

Qanus

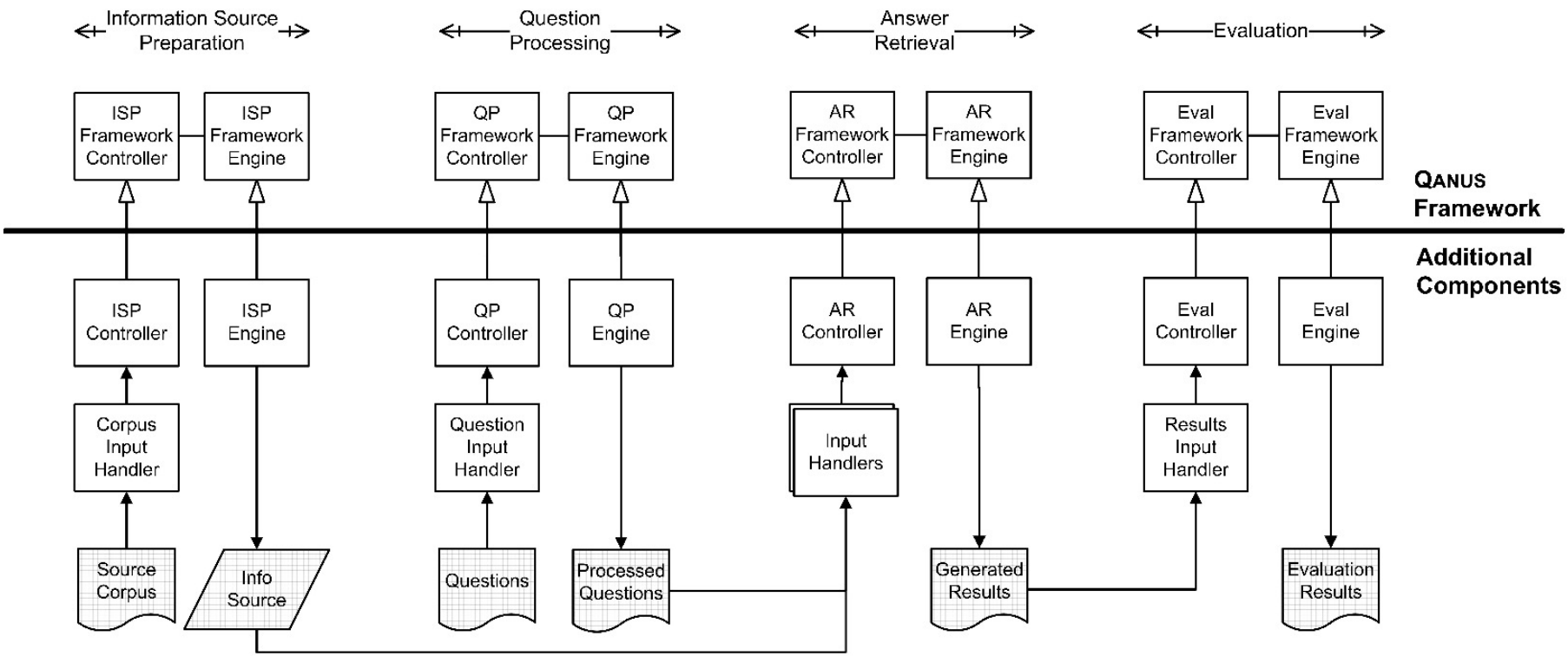


Figura: El framework Quanus y la implementación QA-sys

El pipeline de QA-Sys

1. Preparación de la fuente de información: Preprocesa XML de Aquaint (datos de la TREC 07) de manera offline (llamado previo) en un índice Lucene

2. Análisis de la pregunta: Anota la pregunta con POS, NER y QC.

3. Generación de respuestas: Busca la pregunta en el índice, separa los documentos en pasajes y los rankea con métricas heurísticas y luego, dependiendo del tipo de QC, aplica heurísticas basadas en NER, POS y QC sobre los pasajes para extraer una respuesta

QA-Sys

- Compitió en TREC 07 con buenos resultados:
  - LymbaPA07 (privado) $\rightarrow$ 0.706, 1ero
  - Quanta (universitario) $\rightarrow$ 0.206, 10mo
  - QA-Sys $\rightarrow$ 0.119
- Qué métrica

Baseline: ML QA-Sys

- QA-sys multilingüe monolingüe no crosslingüe.
  - Soporta varios idiomas, pero de a uno a la vez.
- Indice invertido de Wikipedia
- Reemplazamos las herramientas NLP por Freeling y sacamos todo lo dependiente al idioma.

1. Base de Conocimiento

Wikipedia	# Entradas	Nulas	Redirects	Filtradas	Válidas
simple-06	18273	22	3452	5241	9558
es-06	233750	52	62805	34947	135946
pt-07	498039	80	210983	43390	243586
simple-13	180067	8	35600	41902	102557

Cuadro: Wikipedias: cantidad de entradas válidas e inválidas

- Wikipedia $\rightarrow$ Lucene (id, title, body, all)
- Reemplazamos el componente del framework por gwtwiki

2. Anotado de la pregunta

- Preguntas Aquaint, Stanford NER, POS, QC
- Preguntas de CLEF '07, Freeling NER y POS, Stanford QC (en)
  - No hay QC en español ni portugués
  - Tareas en-es y en-pt activas → Preguntas en inglés

Clase	# Preguntas (es)	# Preguntas (pt)
HUM	62	53
NUM	53	52
ENTY	34	28
DESC	25	30
LOC	24	36
ABBR	2	1

Cuadro: Distribución por clases de las preguntas para español y portugués (Li & Roth, Stanford)

3. Generación de Respuestas

- Sacamos Stanford y pusimos Freeling.
- Eliminamos reglas basadas en el inglés.

El algoritmo tiene los siguientes pasos:

1. Filtrado de preguntas
2. Generación de Queries
3. Information Retrieval
4. Extracción y rankeo de oraciones
5. POS tagging de las mejores oraciones
6. Heurísticas de AR basadas en el QC

3.1 Filtrado de preguntas

- Descarta preguntas no *factoid*.
- 32 de tipo *definition* y 10 de tipo *list* para español (24 y 9, respectivamente, solo wikipedia) y 31 *definition* y 10 *list* para portugués (18 y 8, para wikipedia).

3. Generación de Respuestas (2)

3.2 Generación de Queries

- QA-Sys tiene 2 algoritmos:
- 1. Para 'HUM:ind': expresiones regulares escritas a mano (en inglés):
  - "Who (is—was) (the) NN ... NN?" (NN = sustantivo). Por ejemplo: "Who is the CEO of Microsoft?".
  - "Who VERBO ...?".
  - Eliminamos este caso, depende de inglés.
- 2. *FormQuery*. No 'HUM:ind'
  - Espera 'target' (TREC). Lo vinculamos con el tópico de CLEF (ver más adelante) .
  - Todas las palabras no-stopwords del campo *target* y todos los sustantivos, verbos y adjetivos de la pregunta completa.
  - Este método adaptado a Freeling es nuestro baseline.
  - Usos de target:

3. Generación de Respuestas (3)

3.3 IR

- Pedido a Lucene y lista de N documentos.

3.4 Extracción y rankeo de oraciones

- Documentos divididos en oraciones (no heredan rank)
- Peso ponderado por features.
- $PassageScore = (0{,}9) * CoverageScore + (0{,}05) * FreqScore + (0{,}05) * ProxScore;$
- Scorers entre 0 y 1, ver próxima slide

3.5 POS tagging de las mejores oraciones

- Se aplica POS tag a las 40 oraciones mejor rankeadas según *PassageScore*
- Se desecha el resto

Scorers: Freq, Cov, Prox, Span

Freq & Cov: Ocurrencia de la query en la respuesta candidata

- Frecuencia: tokens de la query que ocurren en la oración sobre tokens de la oración
- Cubrimiento: tokens de que query que ocurren en la oración sobre tokens de la query

Prox: distancia de dos strings en un tercero

- s_1 =“Argentina es un país americano” y s_2 =“independizado en 1810”
- t =“Argentina es un país americano, originalmente una colonia española, independizado en 1810”
- $prox(s_1, s_2, t) = fraqdist(s_1, s_2)len(t) = fraq712 = 0{,}58$,
  - $dist(s_1, s_2)$ = cantidad de strings entre ‘un’ y ‘en’ (tokens intermedios de $s_{1,2}$)
  - $len(t)$ = cantidad de strings del texto
  - 1 denota que los dos string están cercanos uno al otro en el tercer string.

Heurísticas de AR - Casos

3.6 Heurísticas de AR basadas en el QC

- Extracción de respuestas en base al tipo de pregunta generado por el clasificador en el paso anterior.
- Casos: ABBR:exp, ABBR:abb, HUM:gr y ENTY:cremat, HUM:ind, HUM general, LOC general, NUM:date, NUM:period, NUM:count, NUM general, ENTY general.

Caso ABBR:exp - expansión de una abreviación

- Se extraen las entidades nombradas de tipo 'Organización' desde la pregunta.
- Se generan regex de expansión de esa abreviación. Por ejemplo, "IGFA" $\rightarrow$ $[I][A-Za-z0-9][G][A-Za-z0-9][F][A-Za-z0-9][A][A-Za-z0-9]$.
- Busca este patrón en las 40 oraciones, devuelve el primer match.

Caso ABBR:abb - contracción de una abreviación

- Por ejemplo: "¿Con qué siglas se conoce a la corporación 'International Business Machines'?" $\rightarrow$ IBM
- El caso no tiene código: *// Abbreviations : Contractions -¿ What can we do?*

Casos - HUM:gr y ENTY:cremat

Caso HUM:gr y ENTY:cremat - grupos y objetos humanos

- HUM:gr refiere a grupos humanos, como compañías u organizaciones,
- ENTY:cremat refiere a entidades de creación humana (como libros, inventos, discos, etc).
- Extraer nombres propios (proper nouns, NNPs) de todas las oraciones rankeadas
  - Nombres propios consecutivos se agrupan (Tiger Woods)/NNP
  - Respuestas candidatas
- Filtro candidatas que contengan palabras en diccionario. [3]
- Re ranking de las candidatas según COMENTADO:
- En la adaptación eliminamos el filtrado por diccionarios ya que depende del idioma

[1] En concreto: {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday}, {January, February, March, April, May, June, July, August, September, October, November, December}, y {us, uk, france, england, cuba, japan, u.s, america}.

Casos - HUM:gr y ENTY:cremat

Caso HUM:ind - individuo humano

- Respuestas candidatas = Extraer NER tipo 'PERSON' de las oraciones.
- Score: $(0{,}5 * \mathit{CoverageScoreTarget}) + (0{,}25 * \mathit{CoverageScoreSubject}) + (0{,}35 * \mathit{SentenceScore}) + (0{,}25 * \mathit{ProximityScore}) + (0{,}1 * \mathit{RepeatedTermScore}) + (0{,}5 * \mathit{IsPronoun})$
  - CoverageScoreTarget: cuántos tokens del 'target' aparecen en la oración fuente de la respuesta candidata
  - CoverageScoreSubject: cuántos tokens del 'subject' aparecen en la oración fuente de la respuesta candidata (si se pudo encontrar un subject, cero si no - en nuestra adaptación es siempre cero)
  - SentenceScore: puntaje derivado de la posición de la oración-fuente en el ranking de oraciones
  - ProximityScore: cuán cerca están la query utilizada como input del módulo de information retrieval (sin stop-words) y la respuesta candidata en el contexto de la oración de la que se extrajo la respuesta candidata.
  - RepeatedTermScore: penalización (negativo). Coverage entre el 'target' y la respuesta candidata
  - IsPronoun: Penaliza si la respuesta candidata contiene pronombres

Casos

Caso HUM general

Este caso contempla todos los tipos de respuestas esperadas de clase HUM (humano) que no son gr ni ind (estos son dos casos: título y descripción). En este caso, se toma el primer nombre propio de la oración mejor rankeada y se utiliza eso como respuesta.

Caso LOC general

La clase LOC (location o locación) incluye preguntas que refieren a lugares. En este caso, se extraen todas las entidades nombradas de tipo 'LOCATION' de las oraciones rankeadas y se las evalúa según el siguiente scoring: $TotalScore = (0{,}6 * CoverageScore) + (0{,}1 * SentenceScore) + (0{,}2 * ProximityScore) + (0{,}5 * SanityScore) + (0{,}3 * RepeatedTermScore)$

Donde los scores representan lo siguiente:

- CoverageScore: cuántos tokens del 'target' aparecen en la oración fuente de la respuesta candidata
- SentenceScore: puntaje derivado de la posición de la oración-fuente en el ranking de oraciones
- ProximityScore: cuán cerca están la query utilizada como input del módulo de information retrieval (sin stop-words) y la respuesta candidata en el contexto de la oración de la que se extrajo la respuesta candidata

Casos

Caso NUM:date

Este tipo de respuesta esperada representa una fecha. El algoritmo verifica la ocurrencia del término 'year' en la pregunta. Si la pregunta contiene el término, genera un patrón para buscar años (una expresión regular) y devuelve el primero que encuentra. En caso contrario, devuelve el título del primer documento encontrado (ya que el módulo está implementado para responder preguntas de TREC y los documentos son artículos cuyos títulos tienen, en general, fechas). Como oración de justificación, toma la primer oración del documento.
En nuestra adaptación, recaímos en el caso general para la clase NUM, especificándole a Freeling que solo considere fechas (ver tres títulos más abajo).

Caso NUM:period

Este tipo de respuesta esperada representa un periodo de tiempo. El código busca en la pregunta por el patrón "How old (is—was) NNP1,?" (donde "NNP1," significa uno o más nombres propios seguidos) y se queda con la secuencia de nombres propios como *subject*. Si logra detectarse este subject, se busca en las oraciones rankeadas por números y por números seguidos de los tokens "years old". Luego se rankean estos números según la siguiente fórmula:

Casos

Caso NUM:count

Este tipo de respuesta esperada representa un número. El algoritmo busca por el patrón "How many ...?" intentado extraer el *subject*, por ejemplo, para la pregunta "How many miners...?" el subject es "miners". Con este subject, realiza exactamente los mismos pasos que el caso inmediatamente anterior y nosotros realizamos la misma adaptación, esta vez especificando que los números buscados no sean fechas.

Caso NUM general

Considera los casos numéricos no contemplados en los tres casos anteriores. El algoritmo ejecutado consiste en encontrar el primer set continuo de tokens taggeados como 'CD' (números) por el POS tagger de Stanford. En nuestra adaptación a Freeling, pudimos mejorar esto separado dos tipos de números: los referidos a fechas y los referidos a cantidades. En el caso general, se busca cualquier tipo de números. Sin embargo, como especificamos en los títulos inmediatamente anteriores, para ciertos casos especificamos o bien fechas, o bien cantidades.

Caso ENTY general

El caso de entidades general incluye todas las subclases que no son 'cremat'. El algoritmo implementado es el mismo que para el caso de 'HUM' general,

Modificaciones

- Inferencia del tópico del grupo de preguntas
- Generación de queries
- Generación de respuestas

Inferencia del tópico del grupo de preguntas

- Grupos de 1 a 4 preguntas con tema
- Ejemplos: Colegio de Harry Potter, Pez Espada, Revolución de Terciopelo
- Condiciones del tema
  - Nombrado en primer pregunta/respuesta
  - Las siguientes preguntas pueden contener correferencias a este tópico
- Incorporado al sistema como "Target"
- Probamos diferentes valores:
  1. Test (el del test set)
  2. NERs + Numeros + Fechas
  3. Sustantivos
  4. 2 y 3
- 2, 3 y 4 basados solo en la pregunta 1.

Generación de queries

1. Baseline / Qanus:
   - Tokens no repetidos ni stopwords de target (tópico)
   - Sustantivos, verbos y adjetivos de la pregunta completa
   - + números + fechas
2. Baseline "mejorado": *(TITLE: target)*^{n} *OR* ***query baseline***, $n = 5$
3. Lasso: como describimos en Estado de Arte
   1. Todas las palabras no stop words entre comillas
   2. Todas las entidades nombradas reconocidas
   3. Todas las construcciones nominales con sus adjetivos
   4. Todas las demás construcciones nominales
   5. Todos los sustantivos con sus adjetivos
   6. Todos los demás sustantivos
   7. Todos los verbos
   8. ~~El focus de la pregunta~~

Generación de respuestas

- Scorers nuevos para rankear los pasajes
- No tocamos las heurísticas
- Scorers:
  - LengthScore : Contempla la longitud del pasaje, priorizando oraciones cortas pero no demasiado cortas. Este score trata de evitar problemas vinculados con mala redacción en los documentos que generaban pasajes enormes y sin sentido para las librerías.
  - QVerbScore: Contempla la presencia de verbos de la pregunta en el pasaje candidato.
  - QNounScore: Contempla la presencia de sustantivos de la pregunta en el pasaje candidato.
  - QNERScore: Contempla la presencia de las entidades nombradas de la pregunta en el pasaje candidato.
- Tres métodos de rankeo de pasajes:
  - Baseline:
    $pscore_{bl} = (0{,}9) * CoverageScore + (0{,}05) * FreqScore + (0{,}05) * ProxScore$;
  - $pscore_2 =$
    $pscore_{bl} * 0{,}4 + QNERScore * 0{,}2 + QVerbScore * 0{,}15 + QNounScore * 0{,}25$
  - $pscore_3 = pscore_{bl} * 0{,}4 + LengthScore * 0{,}15 + QNERScore * 0{,}1 +$
    $QVerbScore * 0{,}1 + QNounScore * 0{,}25$

Experimentos

Corridas

- Idioma: español, portugués (2 opciones)
- Cantidad de Documentos retornados por el módulo de IR: 50, 100, 200 (3 opciones)
- Cantidad de Pasajes extraídos de esos documentos: 20, 40, 70 (3 opciones)
- Método de Generación de Queries: baseline (1), improved baseline (2), lasso (3) (3 opciones)
- Método de Inferencia de Temas: Test (1), NERs (2), sustantivos (3), híbrido (4) (4 opciones)
- Método de Ranking de Pasajes: baseline (1), 2, 3 (3 opciones)

Evaluación de resultados

- Cantidad de respuestas dadas por el sistema: 1, 5, 10, 25 (4 opciones)
- Forma de evaluación automática: exacto, cubrimiento del 100 %, 75 % y 50 %, 15 % y 1 % (6 opciones)

Corrida 1: instanciación de cantidades

Variables variables

- # docs retornados por Lucene $= \{50, 100, 200\}$
- # pasajes extraidos por documento $= \{20, 40, 70\}$

Variables fijas

- Idioma: Español
- Método de Generación de Queries: 1 (baseline)
- Método de Temas: 2 (NERs propio)
- Método de Ranking: 1 (baseline)

Corrida 1: resultados

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	7.69	7.69	8.46	10.00	10.00	10.00
MRR_5	8.87	9.18	9.95	**12.51**	**13.47**	**13.54**
MRR_{10}	**9.37**	**9.44**	**10.21**	**12.77**	**14.25**	**14.31**
MRR_{25}	**9.56**	**9.63**	**10.40**	**13.07**	**14.58**	**14.65**

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	**7.75**	**7.75**	**8.53**	**10.08**	**10.08**	**10.08**
MRR_5	**8.94**	**9.25**	**10.03**	12.42	13.39	13.45
MRR_{10}	9.31	9.38	10.16	12.55	14.03	14.10
MRR_{25}	9.54	9.61	10.39	12.89	14.41	14.48

Cuadro: Corrida 1: con 50 documentos de Lucene y 20, 40 pasajes

Conclusiones

- 9 ejecuciones
- Más documentos de Lucene, peor en todas las permutaciones
- Muchos pasajes (70) también es malo en general
- 20 y 40 se comportan raro.
- 40 es mejor para evaluaciones más exactas
- Fijamos 50 y 40 para el resto de las corridas

Corrida 2.1: Generación de Queries

Variables variables

- Método de generación de queries: Baseline, Baseline "mejorado", Lasso
-

Variables fijas

- Español, 50 documentos, 40 pasajes
- Método de Temas: 2 (NERs propio)
- Método de Ranking: 1 (baseline)

Conclusiones

- Método 3 (Lasso) ligeramente superior a baseline
- Baseline "mejorado" rezagado siempre

Corrida 2.1: Resultados

1) Baseline

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	7.75	7.75	8.53	10.08	10.08	10.08
MRR_5	8.94	9.25	10.03	12.42	13.39	13.45
MRR_{10}	9.31	9.38	10.16	12.55	14.03	14.10
MRR_{25}	9.54	9.61	10.39	12.89	14.41	14.48

2) Improved Baseline

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	3.91	3.91	3.91	3.91	3.91	3.91
MRR_5	5.36	5.91	5.91	8.50	9.31	9.31
MRR_{10}	5.96	6.43	6.43	9.02	10.18	10.18
MRR_{25}	5.96	6.43	6.49	9.20	10.46	10.46

3) Lasso

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	7.81	7.81	8.59	10.16	10.94	10.94
MRR_5	9.27	9.58	10.76	13.52	14.49	14.56
MRR_{10}	9.64	9.71	10.89	13.65	15.15	15.21
MRR_{25}	9.94	10.01	11.18	14.05	15.59	15.65

Cuadro: Corrida 2.1: Generación de Queries 1, 2 y 3 respectivamente

Corrida 2.2: Inferencia de Temas

En esta corrida variamos únicamente el método de inferencia de temas, generando un total de 4 ejecuciones y dejando fijos los siguiente parámetros:

- Idioma: Español
- Cantidad de documentos retornados: 50
- Cantidad de pasajes extraídos: 40
- Método de Generación de Queries: 1
- Método de Ranking de Pasajes: 1

Conclusiones: El método utilizado (NERs + números de la primer pregunta como target) es el más efectivo en todos los casos.

Corrida 2.2: Resultados

1) Tema Test

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	6.15	6.15	6.15	6.92	6.92	6.92
MRR_5	7.87	8.41	8.79	10.21	10.85	10.85
MRR_{10}	8.08	8.49	8.87	10.53	11.77	11.77
MRR_{25}	8.29	8.72	9.16	10.91	12.18	12.18

2) Tema NERs

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	7.75	7.75	8.53	10.08	10.08	10.08
MRR_5	8.94	9.25	10.03	12.42	13.39	13.45
MRR_{10}	9.31	9.38	10.16	12.55	14.03	14.10
MRR_{25}	9.54	9.61	10.39	12.89	14.41	14.48

3) Tema Sustantivos

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	3.85	3.85	3.85	4.62	4.62	4.62
MRR_5	4.19	4.19	4.45	5.88	6.14	6.14
MRR_{10}	4.19	4.30	4.56	5.99	6.43	6.43
MRR_{25}	4.47	4.58	4.83	6.46	7.08	7.08

4) Tema NERs + Sustantivos

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	5.47	5.47	6.25	7.03	7.03	7.03

Corrida 2.3: Ranking de Pasajes

En esta corrida variamos únicamente el método de ranking de pasajes, generando un total de 3 ejecuciones y dejando fijos los siguiente parámetros:

- Idioma: Español
- Cantidad de documentos retornados: 50
- Cantidad de pasajes extraídos: 40
- Método de Generación de Queries: 1
- Método de Temas: 2

Corrida 2.3: Aclaraciones

Los resultados pueden observarse en la tabla 9. Las fórmulas nuevas propuestas no resultaron de utilidad en ningún caso. Viendo la diferencia de resultados entre los métodos 2 y 3, se hace claro que el *LengthScore* aporta, ya que es el único agregado sustantivo en el método 3 y este tiene una mejoría significativa en comparación con el 2. De allí surgió la idea de aplicarlo a la fórmula baseline. Hicimos dos modificaciones: Una ponderando 90 / 10 y otra ponderando 80 / 20 y 70 / 30 dando buenos resultados las tres (Ver tabla 10). La fórmula del score es la siguiente:

$$LengthScore = \begin{cases} 1{,}0 & 4 < \#tokens < 100 \\ 0{,}5 & 100 \leq \#tokens < 200 \\ 0{,}0 & \text{En cualquier otro caso} \end{cases}$$

Como se puede ver en 10, agregar este valor en el score general del pasaje mejora los resultados. Las causas más evidentes son que el score funciona como un filtro para pasajes mal formados (es decir, que por alguna razón el splitter no logró 'separar' bien) o bien pasajes bien formados pero excesivamente largos. En ambos casos, las herramientas de procesamiento de lenguajes pierden su performance, por lo que pasajes bien rankeados, en el momento de la extracción de la respuesta no son de utilidad. Evaluamos tres ponderaciones -9/1, 8/2 y 7/3-, dando mejores resultados cuanto más se ponderó el score, salvo para MRR_1 que se redujo en la tercer corrida. Dado

Corrida 2.3: Resultados 1

	$PassageScore_{bl}$					
Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	7.75	7.75	8.53	10.08	10.08	10.08
MRR_5	8.94	9.25	10.03	12.42	13.39	13.45
MRR_{10}	9.31	9.38	10.16	12.55	14.03	14.10
MRR_{25}	9.54	9.61	10.39	12.89	14.41	14.48

	$PassageScore_2$					
Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	3.10	3.10	3.10	4.65	4.65	4.65
MRR_5	5.06	5.22	5.22	7.51	8.35	8.41
MRR_{10}	5.35	5.50	5.50	7.89	9.18	9.24
MRR_{25}	5.39	5.55	5.55	8.10	9.60	9.67

	$PassageScore_3$					
Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	5.38	5.38	5.38	6.15	6.15	6.15
MRR_5	6.69	6.85	7.10	9.35	10.69	10.88
MRR_{10}	7.15	7.31	7.56	9.98	11.66	11.85
MRR_{25}	7.19	7.35	7.60	10.21	11.99	12.19

Cuadro: Corrida 2.3: Fórmulas 1, 2 y 3 respectivamente

Corrida 2.3: Resultados 2

$PassageScore_{bl} \times 0{,}9 + LengthScore \times 0{,}1$

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	7.81	7.81	8.59	10.16	10.16	10.16
MRR_5	9.47	9.78	10.56	12.97	13.95	14.14
MRR_{10}	9.71	9.78	10.56	12.97	14.47	14.66
MRR_{25}	10.01	10.08	10.86	13.38	14.92	15.11

$PassageScore_{bl} \times 0{,}8 + LengthScore \times 0{,}2$

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	**10.08**	**10.08**	**10.85**	**12.40**	**13.18**	**13.18**
MRR_5	11.68	11.99	13.15	15.80	16.58	16.77
MRR_{10}	11.92	11.99	13.15	15.80	16.99	17.18
MRR_{25}	12.27	12.34	13.51	16.26	17.55	17.74

$PassageScore_{bl} \times 0{,}7 + LengthScore \times 0{,}3$

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	10.00	10.00	10.77	12.31	13.08	13.08
MRR_5	11.94	12.50	13.65	17.04	17.81	18.00
MRR_{10}	12.35	12.67	13.83	17.12	18.32	18.51
MRR_{25}	12.58	12.91	14.06	17.46	18.70	18.90

Cuadro: Corrida 2.3: Combinación métodos 1 y 3

Corrida 3: Combinación de Óptimos

Datos de la corrida 3.1, para español:

- Idioma: Español
- Cantidad de documentos retornados: 50
- Cantidad de pasajes extraídos: 40
- Método de Generación de Queries: 3
- Método de Temas: 2
- Fórmula de Ranking de Pasajes: $PassageScore_{bl} \times 0{,}8 + LengthScore \times 0{,}2$

Datos de la corrida 3.2, para portugués:

- Idioma: Portugués
- Cantidad de documentos retornados: 50
- Cantidad de pasajes extraídos: 40
- Método de Generación de Queries: 3
- Método de Temas: 2
- Fórmula de Ranking de Pasajes: $PassageScore_{bl} \times 0{,}8 + LengthScore \times 0{,}2$

Corrida 3: Notas

En esta última corrida, utilizamos los métodos que mejores resultados dieron, por sección, todos juntos, primero para español y luego para portugués. Hay dos observaciones importantes. Primero, con respecto a la corrida en español, es notorio como el MRR_1 y, en general, todos los valores salvo el MRR_5 con matching exacto, tienen una performance peor que en la corrida 2.3, lo cual se debe, seguramente, a una interacción compleja entre los distintos pasos que conduce a este comportamiento.
En segundo lugar, con respecto a la corrida en portugués, es notoria su baja performance en relación con los resultados en español, obteniendo apenas un 3.16 considerando la métrica MRR_5. Existen varios factores para explicar esto. En primer lugar, que las herramientas para portugués no son tan maduras como las herramientas para español; en segundo lugar, que la evaluación de los métodos óptimos y, en general, todo el enfoque al sistema, estuvieron marcados por el español, siendo las adaptaciones para portugués un momento secundario que intentó mostrar, simplemente, que este soporte estaba accesible; en tercer lugar, también es posible que el sistema per se y el enfoque, aún sin otro inconveniente, se desempeñe peor para el set de preguntas para el portugués. A fin de cuentas, no hay que perder de vista que los sets de test usables por nuestro sistema son de 130 preguntas para español y 104 para portugués, a partir de los cuales no es posible extrapolar un comportamiento general, sino presentar resultados concretos.

Corrida 3: Resultados

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	10.00	10.00	10.77	12.31	13.08	13.08
MRR_5	**12.00**	12.31	13.46	16.18	16.95	17.01
MRR_{10}	12.24	12.31	13.46	16.26	17.44	17.50
MRR_{25}	12.55	12.62	13.78	16.72	18.00	18.06

Cuadro: Corrida 3.1: Combinación de óptimos para español

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	1.92	2.88	2.88	3.85	3.85	3.85
MRR_5	3.16	5.03	5.03	6.91	7.15	7.15
MRR_{10}	3.45	5.33	5.33	7.33	7.71	7.86
MRR_{25}	4.08	6.00	6.00	8.21	8.84	8.94

Cuadro: Corrida 3.2: Combinación de óptimos sobre portugués

Limitaciones y trabajo futuro

- Módulo multilingüe de Question Classification.
- Inferencia del tópico a partir del primer par de pregunta-respuesta.
- Generación del *focus*. Tema aparte.
- Preguntas de tipo definition y list, booleanas, cláusulas temporales.
- Mejorar las heurísticas

Conclusiones: Dominio abierto

- Sistema de question answering de dominio abierto con soporte para diferentes idiomas
- Qanus, Qa-sys monolingüe + Freeling + Lasso + Scorers
- CLEF'07 español y portugués + solo wikipedia + solo factoid + diferentes configuraciones
- CLEF'07: la mejor precisión general bajó del 49 % al 41.75 % para tareas multi-lingües, mientras que, más significativamente, bajó de 68 % a 54 % en tareas monolingües
- nos restringimos a una subsección sencilla (eliminando preguntas de tipo DEFINITION y LIST), por lo que no es del todo válido comparar esta performance con la nuestra.
- Qasys, el sistema baseline que adaptamos para soportar diferentes idiomas, sabemos que en la TREC 07 (en inglés), competencia en la que Qasys participó, el mejor sistema, LumbaPA07, obtuvo una exactitud de 70,06 %, mientas que el décimo (Quanta) obtuvo 20,6 % y Qasys logró un 11,9 %.
- Nuestro sistema adaptado al español, con todas las mejoras en su óptimo, logra un nada despreciable 10,08 para MRR_1 y un 12,00 para MRR_5.

Conclusiones 2

Sobre modelo de dominio abierto:

- Limitación de las mejoras a zonas acotadas y sin enfoques sistémicos.
- Una mejora argumentada y con cierta presencia en la literatura del área fue el mecanismo de generación de queries que llamamos Lasso (por el sistema en el que fue implementado por primera vez) y nosotros lo incorporamos aquí,
  - mejoras mínimas en la performance, de 7.75 a 7.81 en MRR_1 exacto ($+0.26$), de 8.94 a 9.27 en MRR_5 exacto ($+0.31$).
- Falta de "doctrina."[en] la generación de respuestas. Oportunidad! Linea muerta?
- Módulo de *focus* open source en inglés. Problema acotado, bien definido. Herramienta útil

index.tex

Conclusiones: Dominio cerrado

- FILL ME

¿Preguntas?