Universidad Nacional de Rosario

Tesina de Licenciatura en Ciencias de la Computación

Sistema de búsqueda con expansión de la consulta para la recuperación de documentos legales.

Joel Arnaldo Catacora Gimenez

supervisado por
Casali, Ana
Deco, Claudia

Departamento de Ciencias de la Computación
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Av. Pellegrini 250, Rosario, Santa Fe, Argentina

Resumen

La búsqueda de documentos en ámbito legal es de vital importancia para el buen desempeño de las tareas que realizan los profesionales de la ley, por ejemplo la construcción del marco legal de un caso jurídico requiere leyes, jurisprudencia y doctrina que debe recolectar un abogado. Se toma como objeto de análisis el Sistema Argentino de Información Jurídica (SAIJ), un sistema nacional de recuperación de documento legales. Una de sus principales características es la búsqueda facetada. La búsqueda por facetas es un paradigma popular en aplicaciones Web, como el comercio electrónico, que le permiten al usuario navegar sobre datos multidimensional. Sin embargo, este enfoque puede generar un proceso *drill-down* de facetas no óptimo. Como alternativa, se propone un sistema de búsqueda semántico, el cual expande la consulta del usuario mediante la retroalimentación por relevancia. Para esto se utilizan las entidades o conceptos pertenecientes a una base de conocimiento como medio para reformular la consulta. Se muestran dos modelos de expansión de la consulta, uno automático y otro interactivo, este último le sugiere al usuario conceptos relacionados a su búsqueda inicial. El sistema de búsqueda propuesto se prueba en el ámbito legal, a partir de dos fuentes de información: un subconjunto de los sumarios del Sistema Argentino de Información Jurídica (SAIJ) y el Tesauro SAIJ del Derecho Argentino. Con este sistema se pretende reducir el esfuerzo de búsqueda que debe realizar el usuario para hallar los documentos que satisfagan su necesidad de información.

Agradecimientos

Para mí son muchos los actores sociales que han contribuido al proceso de elaboración de la escritura de esta tesina para su finalización y la culminación de la carrera de Licenciatura en Ciencias de la Computación, en la histórica y prestigiosa Facultad de Ciencias Exactas, Ingeniería y Agrimensura de la Ciudad de Rosario (Santa Fe, Argentina). En primer lugar, quiero agradecer a las doctoras, Ana y Claudia que me guiaron por una serie de años en este trabajo de investigación, y al doctor Fernando Roda por sus consejos. En segundo lugar, agradezco a los abogados y amigos expertos que brindaron sus conocimientos legales para la conclusión de esta tesina, a continuación diré sus nombres: Diego, Angie, Manuel, Yanina, Eugenía, Mario y Marina. También, le doy las gracias a Luciano Perezzini, sin el apoyo de sus redacciones no hubiese conseguido este resultado. Por último, le doy un agradecimiento a mi familia, integrada por mi madre Gila y por mi hermana Estefanía.

Índice general

Capítulo 1

Introducción

La Recuperación de Información consiste en mostrarle al usuario documentos relevantes frente a una consulta de palabras claves. Su aplicación más común es la búsqueda en la web, la cual es crucial para aplicaciones de varios dominios: corporativos, gubernamentales, etc.

Por otra parte, se tienen una gran cantidad de datos publicados con estándares de la Web Semántica llamados Datos Enlazados (*Linked Data*). Los Datos Enlazados se encuentran embebidos en páginas HTML o contenidos en datasets en formato RDF, un ejemplo de esto son las bases de conocimiento. Cuando estos datos son abiertos, se los llama Datos Abiertos Enlazados (*Linked Open Data*). El gobierno argentino brinda información pública como parte de una iniciativa de Gobierno Abierto, donde se le permiten a los ciudadanos desarrollar sus propias aplicaciones utilizando estos datos para extraer nueva información y conocimiento. A través de un decreto que tiene por objetivo fortalecer la política nacional de apertura de datos públicos, se instruye a los organismos del Poder Ejecutivo a identificar bases de datos a abrir y publicarlas en el portal Datos Argentina[1].

La disponibilidad de datos estructurados ha sido aprovechado para mejorar a los modelos de búsqueda tradicionales. Las entidades o conceptos pertenecientes a una base de conocimiento pueden incorporarse en un sistema de búsqueda para mejorar el entendimiento de las intenciones del usuario, su consulta y los documentos, más allá del rendimiento que pueden alcanzar las palabras o *tokens* por su cuenta. En la literatura, este tipo de mecanismo recibe el nombre de búsqueda semántica: "La búsqueda semántica se compone de una variedad de métodos y enfoques destinados a asistir al usuario en el acceso de su información y el consumo de actividades, mediante el entendimiento de su contexto e intenciones" [9, p. 16].

Uno de los enfoques utilizados para incorporar en los modelos de búsqueda la información de una bases de conocimiento es mediante la expansión de la consulta. Este mecanismo de reformulación de la consulta consiste en añadir a la consulta original otras palabras que capturen la intención del usuario o simplemente produzcan una consulta más útil, es decir, hacer más probable la recuperación de documento relevantes. Una los problemas que pretende solucionar es la discordancia de términos. Esto sucede cuando las palabras que utiliza el usuario no coinciden con los términos de documentos que son relevantes, a causa de la sinonimia y polisemia. Por lo tanto, los recursos pertenecientes a una base de conocimiento, llamados entidades o conceptos, puede ser un medio para expandir la consulta del usuario.

En esta tesina, se propone expandir la consulta utilizando la retroalimentación por

[1] `https://datos.gob.ar/`.

relevancia incorporando la información semántica disponible en una base de conocimiento. Esta técnica en lugar de expandir la consulta con los términos de los documentos relevantes utiliza los términos de las entidades que se encuentran en una base de conocimiento.

Se evalúan los algoritmos de búsquedas propuestos en el ámbito legal, donde encontrar lo que usuario desea es decisivo para su buen desempeño laboral. por ejemplo, elaborar una estrategia de defensa. Se lo aplica en un escenario donde se utilizan a los tesauros para la recuperación de información legal. Por ello, se tienen dos fuentes de información disponibles:

- Un tesauro del dominio.
- Una colección de documentos, indexados temáticamente con el tesauro.

Las aplicaciones comunes para estos datos son las búsquedas por facetas y las expansiones de la consulta mediante el uso del tesauro (añadiendo por ejemplo sinónimos). Aquí, se combinan estas fuentes de información para aplicar el sistema de búsqueda propuesto.

Las fuentes de información con los que se realizó la experimentación provienen del Sistema Argentino de Información Jurídica [2] (SAIJ). El SAIJ es una base de datos documental en línea de acceso gratuito, contiene legislación, jurisprudencia, doctrina, tanto nacional como provincial, entre otro tipos de documentos legales. Es administrado por el Ministerio de Justicia y Derecho Humanos de la Nación Argentina. SAIJ ofrece búsquedas por facetas a partir del Tesauro SAIJ del Derecho Argentino[3]. El usuario puede ingresar como consulta un tema del tesauro del SAIJ para recuperan los documentos que se encuentran clasificados con dicho tema. Encontramos ciertas limitaciones en la búsqueda por facetas y las búsquedas por palabras claves, por ejemplo el usuario es el encargado de hallar los temas más cercanos a su necesidad de información, es decir, no se proveen sugerencias, y solo pueden expresarse búsquedas por palabras claves estrictamente conjuntivas.

En este trabajo, se utilizan dos modelos de retroalimentación por relevancia a partir de una base de conocimiento construida sobre los datos del tesauro del SAIJ y los documentos del SAIJ indexados temáticamente. De esta manera, se logra utilizar toda la información semántica disponible para mejorar la precisión de los resultados y la experiencia de búsqueda del usuario. Uno de los modelos propuestos provee de sugerencia de búsquedas al usuario. El modelo de expansión interactiva de la consulta le muestra al usuario conceptos de la base de conocimiento relacionados a una búsqueda inicial de palabras claves.

En la Argentina, encontramos investigaciones recientes sobre la búsqueda o recomendación de información legal y el reconocimiento de entidades nombradas en documentos legales, destacamos las siguientes:

- Sistema de soporte para la recuperación de normativas en la ingeniería legal [75]: se propone un sistema de recomendación de normativas para construir de manera semi-automática la matriz legal de una empresa, esta matriz define el conjunto de normas que son de interés para la empresa, es decir, aquellas que regulan su actividad. La necesidad de información del usuario se expresa a partir de temas del tesauro del SAIJ y se recomiendan las normas cuya clasificación automática, mediante el algoritmo *Support Vector Machine*, coincidan con los temas de la consulta.

[2] `http://www.saij.gob.ar/`.

[3] `http://vocabularios.saij.gob.ar/portalthes/?v=1`.

- Modelo de recuperación de información jurídica basado en ontologías y distancias semánticas [28]: Se propone un sistema de búsqueda para la recuperación de sentencias judiciales, se utilizan vocabularios legales y generales (ConceptNet, WordReference, Banco de Vocabularios Jurídicos Argentinos) para expandir la consulta del usuario y ranquear los documentos a partir de la similitudes basadas en *Normalized Google Distance*.

- Mejora del acceso a Infoleg mediante técnicas de procesamiento automático del lenguaje [20]: se identifican entidades legales nombradas en normas nacionales de InfoLeg[4], los tipos o clases de entidades consideradas son las leyes, resoluciones, decretos, entre otros. Como método de Reconocimiento y Clasificación de Entidades se utilizan reglas manuales y al algoritmo supervisado NERC de Stanford.

- Enhancing domain-specific ontologies at ease [19]: se alinea la ontología legal LKIF con la ontología de dominio general YAGO. Se construye un dataset para entrenar algoritmos de Reconocimiento y Clasificación de Entidades Nombradas, utilizando la alineación entre LKIF y YAGO.

- Sistema de recomendación para textos legales [18]: se desarrolla un sistema de recomendación de normativas. Se prueba el Modelo Espacio Vectorial, tanto con vectores densos, generados con el algoritmo word2vec, como dispersos, dados por TF-IDF.

A continuación, se detalla la estructura de la tesina. En los primeros cuatro capítulos se presentan los conceptos preliminares que dan paso a la propuesta de esta tesina. En el Capítulo 1 y 2 se describen los modelos clásicos de la Recuperación de Información, en el Capítulo 3 se describen modelos para la recuperación de un tipo particular de dato: las entidades o conceptos pertenecientes a una base de conocimiento. En el Capítulo 4, se introducen conceptos sobre bases de conocimiento y vocabularios controlados. Mientras que en el Capítulo 5 se detallan los modelos de expansión de la consulta propuestos y en el Capítulo 6 se exponen los resultados de las evaluaciones realizadas sobre colecciones de test propias. Por ultimo, en el Capítulo 7 se encuentran las conclusiones.

1.1. Recuperación de información

La recuperación de información se encarga de hallar material —documentos, imágenes, audio, etc.— de naturaleza no estructurada que satisfaga la necesidad de información del usuario sobre una gran colección, almacenada generalmente en computadoras [58].

En esta tesina se considera como unidad de recuperación el texto proveniente de documentos, más adelante, en la recuperación de entidades, se verá el uso de un tipo especial de documento, llamado descripciones de entidades.

La naturaleza *no estructurada* del texto significa que los documentos no tienen una estructura rígida que facilite su recuperación, en contraste con lo que sucede en las bases de datos. Los datos estructurados generalmente se almacenan en bases de datos relacionales, tabulados y bajo un estricto esquema. La búsqueda es realizada utilizando un lenguaje de búsqueda formal, como SQL. Estos lenguajes permiten una formulación de la necesidad de información muy precisa, pero requieren que el usuario conozca en profundidad el lenguaje y el esquema de la base de datos [9, p. 25].. Sin embargo, un texto no estructurado no significa que no contenga una estructura, como encabezados, párrafos, anotaciones —p. ej., HTML, XML—, etc.

[4] `http://www.infoleg.gob.ar/`.

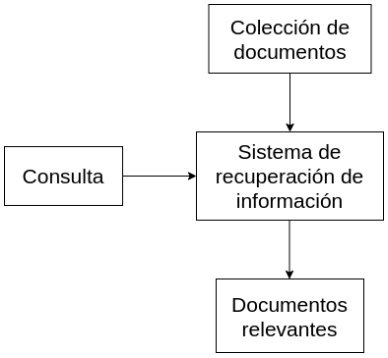


Figura 1.1: Sistema de recuperación de información (Fuente [102])

El proceso de recuperación comienza cuando el usuario ingresa una consulta al sistema, ver Figura 1.1. Los resultados pueden o no satisfacer su consulta, la cual es una descripción de su *necesidad de información*, es decir el/los tema/s de interés para el usuario. Si percibe que el documento recuperado contiene información de valor con respecto a su necesidad de información, se dice que el documento es *relevante*. Para retornar documentos relevantes al usuario, se utilizan modelos de recuperación, estos son una representación formal del proceso de correspondencia entre la consulta y el documento [27]. Los documentos recuperados son ordenados de acuerdo al puntaje asignado a cada uno de ellos por el algoritmo de ranqueo del modelo de búsqueda, se pretende que dicho valor refleje su relevancia ante la consulta. A continuación, iniciaremos un breve repaso histórico por las principales acontecimientos producidos en el campo de la Recuperación de Información, siguiendo el desarrollo de Sanderson y Croft [89].

Los sistema computarizados para la recuperación de información, también llamados motores de búsqueda, se iniciaron en la década de 1940[5]. Ante un gran incremento en la producción de literatura científica junto a la disponibilidad de computadoras, se incrementó el interés en la recuperación automática de documentos. En un principio, la recuperación de documentos se basada en autores, títulos y palabras claves, la búsqueda sobre el texto completo vendría mucho después. El tipo de búsqueda de estos sistemas era la recuperación booleana. La consulta era una combinación lógica de términos, y se obtenían el conjunto de documentos que satisfacían la consulta.

En la década de los 50, Luhn y Maron et al. propusieron otro método de recuperación, donde a cada documento de la colección se le asignaba un puntaje para indicar su relevancia ante una determinada consulta. Los puntajes asignados a los documentos se obtenían de acuerdo a métodos probabilísticos. En una investigación de Luhn se introdujo la frecuencia de un términos como medida de su importancia, lo que posteriormente sería llamado como peso *frecuencia del término*.

Otro enfoque, originado en los 60, consistió en ver a los documentos y a las consultas en un gran espacio dimensional, llamado luego Modelo Espacio Vectorial. En los 70, el trabajo de Spärck Jones introdujo la idea de que la frecuencia de una palabra en una colección de documentos era inversamente proporcional a su importancia en el proceso de recuperación, a esto se lo denominó como *frecuencia inversa del término*. En esta década, se propusieron modelos de recuperación que extendieron la idea de utilizar la teoría de la probabilidad, iniciada por Maron et al., se destaca la investigación de Robertson, quien definió el Principio de Ordenamiento por Probabilidad, lo que determinó como ordenar

[5]La primera referencia a una computadora utilizada para la búsqueda de contenidos es una conferencia de la Royal Society de Londres en 1948. Holmstrom describe una máquina llamada Univac capaz de buscar referencias de texto asociadas con un código de temas. [89]

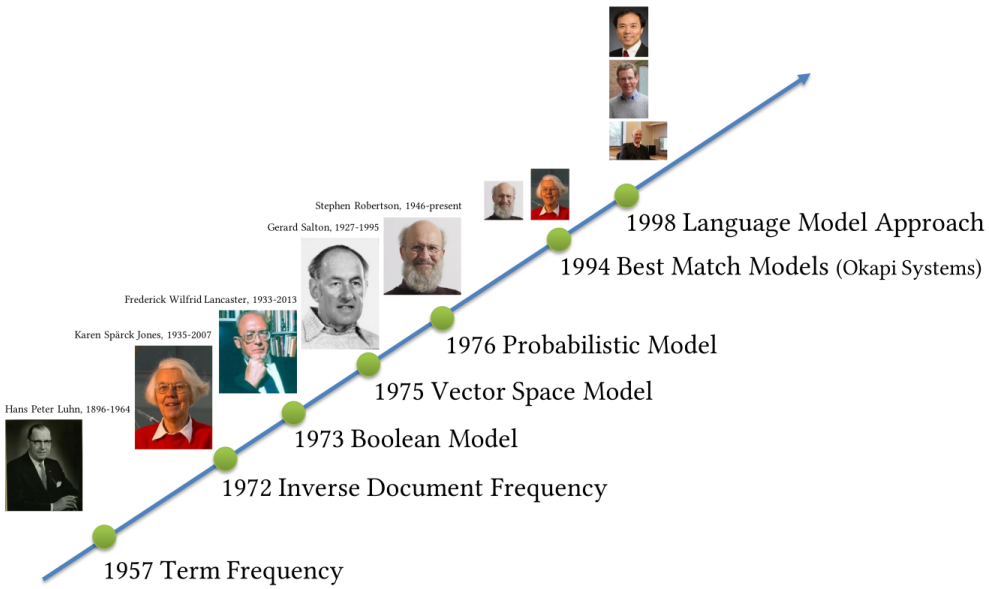


Figura 1.2: Evolución de los modelos de recuperación de información (Fuente [23])

a los documentos de manera óptima de acuerdo a medidas probabilísticas.

En los 80 y 90, se propusieron nuevos esquemas de pesos para el Modelo Espacio Vectorial y continuarían extendiéndose los modelos formales. Se diseñó la función de ranqueo BM25, basado en el modelo probabilístico, que lograría incorporar el peso de la frecuencia del término, algo que el modelo probabilista original no incluía. Además, se desarrolló otro modelo probabilístico basado en los modelos del lenguaje que sería útil para entender diversos procesos de la recuperación de la información, como la retroalimentación por relevancia y la dependencia entre términos [89].

A fines de los 90, los motores de búsqueda en la web se convertirían en la más común y quizás la mejor implementación de los modelos de recuperación de información [58]. En la década de los 2000, se iniciaría con el desarrollo de algoritmos supervisados de ranqueo, gracias a la gran cantidad de datos de entrenamiento disponible en los historiales de búsqueda y a nuevos algoritmos de aprendizaje. Cabe destacar que en esta tesina nos limitamos a utilizar algoritmos de ranqueo no supervisados ya que no tenemos a disposición grandes historiales de búsqueda que permitan entrenar una función de ranqueo.

En la Figura 1.2 se muestran las investigaciones comentadas, las cuales se tratarán en las siguientes secciones y en el Capítulo 2. En la Sección 1.1.1 se comienza describiendo el proceso de indexación que debe realizarse sobre cada documento de la colección, se detalla la trasformación o normalización que sufre el texto en su incorporación al sistema de búsqueda. Luego, se presentan los modelos clásicos de la Recuperación de Información: el Modelo Booleano (Sección 1.1.2), el Modelo Espacio Vectorial (Sección 1.1.3) y el modelo probabilístico BM25 (Sección 1.1.4).

1.1.1. Normalización e indexación de documentos

En la Figura 1.3 se muestra el proceso de indexación, formado por la adquisición del texto, la transformación del texto y la creación del índice [27, p. 14].

La etapa de *adquisición del texto* consiste en identificar y hacer disponible a los documentos que deben ser buscados. En algunos casos, se utiliza una colección ya existente, pero en general se construye una colección propia mediante *crawling* o se aprovecha otras

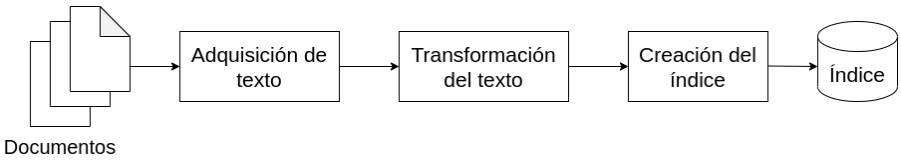


Figura 1.3: Proceso de indexación (Basado en [27, p. 15]).

fuentes de datos.

Luego, la *transformación del texto* convierte a los documentos en términos índice. Estos términos, en general son las palabras que contiene un documento, se almacenan en un índice y luego son utilizados en el proceso de búsqueda. Se los suele llamar simplemente como "términos". El proceso de generación de estos términos se aplica tanto a un documento como a una consulta y se llevan a cabo mediante una serie de transformaciones sucesivas sobre el texto de entrada, denominadas genéricamente operaciones de texto. Esta serie de transformaciones persigue, además, la reducción del texto a algún tipo de forma canónica que facilite el establecimiento de correspondencias durante el posterior proceso de búsqueda. Se suele denominar a este proceso como *normalización* [96, p. 24]. El conjunto de todos los términos que son indexados para una colección de documentos son llamados *vocabulario*.

Por último, la creación del índice toma la salida de la transformación del texto y crea los índices o estructuras de datos que permiten búsquedas más eficientes. El índice invertido es la forma de indexación más utilizada en los motores de búsqueda.

Se detallan, a continuación, las aproximaciones clásicas a las operaciones de texto que incluyen el análisis léxico del texto, la eliminación de las denominadas *stopwords*, la eliminación de mayúsculas y signos ortográficos, el *stemming* de los términos resultantes y, finalmente, la selección de los componentes de los documentos que serán utilizados por el sistema de recuperación [96].

Análisis léxico del texto

El proceso de análisis léxico, o *tokenización*, consiste en la conversión de una secuencia de caracteres —el texto de los documentos o las consultas—, en una secuencia de palabras candidatas a ser adoptadas como *términos índice* por el sistema. Por lo tanto, el objetivo principal de esta primera fase es la identificación de las palabras que conforman el texto.

Dependiendo del dominio de aplicación y las características del idioma, la composición de los diferentes tipos de caracteres y su tratamiento varia. De todos modos, el nivel de complejidad de los *tokenizadores* suele ser escaso, siendo herramientas sencillas y rápidas.

Supresión de palabras vacías

Se denominan palabras vacías, o *stopwords*, a aquellas palabras que son de escasa utilidad, ya que su excesiva frecuencia anula su capacidad discriminante —p. ej., formas verbales de ser o estar—, o su contenido semántico es escaso —p. ej., artículos y preposiciones—. Dada su poca utilidad, dichas palabras son desechadas, lo que permite a su vez un considerable ahorro de recursos, ya que si bien estas palabras representan una parte ínfima del vocabulario, suponen, en cambio, una cantidad muy importante del número de términos a procesar, lo que permite reducir considerablemente el espacio de almacenamiento de las estructuras generadas [96, p. 25]. Sin embargo, hoy en día el costo de incluir las palabras vacías no es muy alto, en términos del tamaño del índice y

el procesamiento de la consulta, por lo que se recomienda indexar a todas las palabras de los documentos [27, p. 91].

Eliminación de mayúsculas y signos ortográficos

El paso de mayúsculas a minúsculas de los términos procesados es habitual en los sistemas de Recuperación de Información, ya que así se puede evitar la falta de correspondencia con términos que, por ejemplo, estén en mayúsculas únicamente por estar al principio de la oración. Es también frecuente que signos ortográficos tales como tildes o diéresis sean eliminados [96, p. 26].

Stemming

En el lenguaje humano un mismo concepto puede ser formulado de diferentes maneras. Cuando se compara un documento con la consulta es posible que aún refiriéndose a conceptos equivalentes, no pueda producirse el *match* necesario debido a la diferencia de sus términos. Por ejemplo, las diferencias en las formas verbales, "cocinar" y "cocinaré", y en las formas nominales, "cocina" y "cocinas". Para minimizar en lo posible el impacto de estos cambios morfológicos en los términos, los Sistemas de Recuperación de Información pueden recurrir a técnicas que permiten el *match* de palabras semánticamente relacionadas, como el *stemming*. Estas herramientas son implementadas mediante los algoritmos denominados *stemmers* [96, p. 26].

El *stemming* reduce las diferentes formas en la que una palabra puede ocurrir debido a la inflexión —p. ej., plurales, tiempos— o a la derivación —p. ej., llevar un verbo a un sustantivo agregando un sufijo—, llevándolo a una raíz común, que presumiblemente contenga la semántica básica del concepto. De esta forma, por ejemplo, los términos anteriores se verían reducidos a la cadena "cocin-", permitiendo de este modo las correspondencias entre los mismos. Si bien el objetivo principal del *stemming* es el de reducir las diferentes formas lingüísticas de una palabra a una forma común o *stem*, y así facilitar el acceso a la información durante el posterior proceso de búsqueda, paralelamente se está reduciendo el número de diferentes términos en el sistema de búsqueda, lo que permite a su vez una segunda reducción de los recursos de almacenamiento requeridos.

El *stemming* incrementa el *recall* (Sección 6.2), pero tiende a reducir la precisión como efecto secundario [58, p. 32]. Además, puede ser particularmente importante para algunos lenguajes, mientras que para otros puede no tener impacto [27, p. 95]. Para el lenguaje español, se mostró que *stemmers* sencillos pueden obtener mejoras significativas [33]. Sin embargo, todos los experimentos demostraron que la naturaleza y el tamaño de la colección de documentos influye directamente en los resultados [67]. En la investigación [68], sobre el uso de *stemming* para la recuperación de jurisprudencia en idioma portugués, se concluye que no todas las colección de documentos legales resultan favorecidas al aplicar esta normalización. Cabe aclarar que la aplicación de los mecanismos de *stemming* no está libre de errores [73].

Entre los algoritmos de *stemming* se destaca el algoritmo de Porter [78] diseñado originalmente para el idioma el inglés, siendo el más utilizado. Este *stemmer* consiste en un número de pasos, cada uno con un conjunto de reglas que se utilizan para remover sufijos. En cada paso, la regla con el sufijo más largo que puede aplicarse es ejecutada. Ha sido adaptado a muchos lenguajes, incluido el español, a partir del *framework* Snowball[6], utilizaremos este *stemmer* en la experimentación.

[6] `https://snowballstem.org/`.

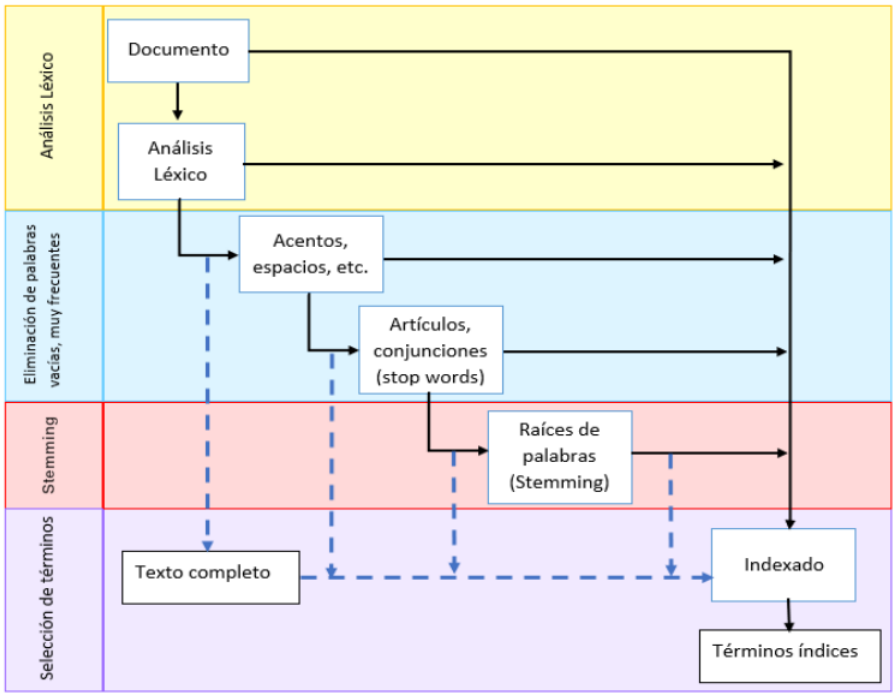


Figura 1.4: Etapas del procesamiento de texto (Fuente [94, p. 11]).

Selección de términos índice

Finalmente, los términos resultantes de las transformaciones de texto previas son adoptados como términos índice [96, p. 27]. Se define así el vocabulario de términos del sistema. Pueden verse las etapas que comprenden el procesamiento de texto para la Recuperación de Información en la Figura 1.4.

En la siguiente sección, se describe el primer modelo desarrollado para la recuperación de información, está basado en la lógica booleana y la teoría de conjuntos, donde tanto los documentos como la consulta del usuario son tratados como un conjunto de términos, esta característica recibe el nombre de bolsas de palabras (*bag of words*).

1.1.2. Modelo booleano

El modelo booleano es el primer modelo de la recuperación de información, en la actualidad sigue siendo usado. En este modelo, una consulta es una expresión booleana, donde los operadores son la intersección (AND), unión (OR) y el complemento (NOT) usuales, y los operandos son los términos del vocabulario.

El conjunto de documentos que se corresponde con los términos de la consulta es combinado de acuerdo a los operadores booleanos. Es decir que el sistema retornará aquellos documentos que pertenezcan al subconjunto expresado por la consulta [64]. El resultado es un conjunto de documentos, que no tiene un orden, pues la consulta establece un criterio específico de recuperación, donde los documentos la satisfacen o no. Por ello, recibe el nombre de *exact-match*.

Como se dijo anteriormente, un documento es visto como un conjunto de palabras, el orden de las mismas no importa, aquellos modelos que tienen esta característica se los llama **bolsa de palabras**, y se aplica en todos los modelos de recuperación que se presentan en esta tesina. Se pueden ver, a continuación, consultas a modo de ejemplo.

Ejemplo 1.1. Se muestra el uso de los operadores AND, OR, y NOT en la generación

de consultas.

- Si queremos especificar que el documento recuperado contenga todos los términos de la consulta, usamos el operador binario AND.

  ```
  motocicleta AND embestimiento
  ```

- En lugar de recuperar los documentos que tienen cada uno de los términos de la consulta, podemos buscar por documentos que contengan al menos uno de los término de la consulta, mediante el operador OR.

  ```
  accidente OR motocicleta
  ```

- A veces, es útil excluir explícitamente ciertos términos mediante el operador NOT.

  ```
  vehículo AND NOT camión
  ```

Este tipo de consultas, le ofrecen al usuarios un mayor control y transparencia sobre los resultados. Pero, requiere de un usuario experto que conozca la colección de documentos y conozca qué busca. Otro inconveniente, es la dificultad de encontrar un balance entre la utilización de los operadores AND y OR. El operador AND tiende a producir una alta precisión, pero baja exhaustividad de los resultados y ocurre lo opuesto con el operador OR [58, p. 15]. Por lo tanto, la cantidad de resultados de las consultas usualmente es mucha o muy poca.

Ante la necesidad de ordenar los resultados se desarrollaron modelos de recuperación que producen un *ranking* de los resultados, es decir que cada documento satisface la consulta de acuerdo a cierto grado de relevancia, se los llama modelos *best-match*. En estos modelos es el propio sistema el que decide cuáles son los documentos que mejor satisfacen la consulta, y donde la consulta pasa a ser *texto libre* [58, p. 13].

Los motores de búsquedas utilizados en la industria, emplean a la búsqueda booleana como un primer filtro para encontrar los resultados que satisfacen la consulta, luego los ordenan de acuerdo a algún modelo *best-match*. En las siguientes secciones, se detallan modelos *best-match*, los cuales establecen una forma de calcular la relevancia de un documento ante una consulta.

1.1.3. Modelo espacio vectorial

La representación de un conjunto de documentos como vectores en un espacio vectorial es conocida como Modelo Espacio Vectorial (MEV), es fundamental para distintas áreas de la ciencias de la computación, como la Recuperación de Información, el Procesamiento del Lenguaje Natural, Aprendizaje Automatizado, y en tareas tales como la clasificación documentos, minería de texto, etc.

Es un modelo eficiente y efectivo que ha sido desarrollado a través de años de experimentación. Sin embargo, como modelo de recuperación, tiene ciertas desventajas, provee pocos detalles de cómo los esquemas de pesos y el algoritmo de ranqueo se relacionan con la relevancia [27, p. 238].

En este modelo, los documentos y la consulta son parte de un espacio k dimensional, donde k es el número de términos del vocabulario. Un documento D_i es representado por un vector donde cada componente se corresponde a un término:

$$D_i = (d_{i,1}, d_{i,2}, ..., d_{i,k}),$$

donde $d_{i,j}$ representa el peso del término j-ésimo. Una colección con n documentos se puede representar como una matriz de pesos, donde cada fila representa un documento y cada columna describe los pesos asignados al término del documento en particular [27, p. 238]:

	Término_1	Término_2	$\dots$	Término_k
D_1	$d_{1,1}$	$d_{1,2}$	$\dots$	$d_{1,k}$
D_2	$d_{2,1}$	$d_{2,2}$	$\dots$	$d_{2,k}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	
D_n	$d_{n,1}$	$d_{n,2}$	$\dots$	$d_{n,k}$

En la siguientes secciones se detalla el esquema de pesos tf-idf y el algoritmo de ranqueo de este modelo.

El esquema de peso tf-idf

Si quisiéramos computar un puntaje que denote la importancia que tiene un término t de la consulta en un documento d, podríamos utilizar como peso el número de ocurrencias de t en d. A este peso se lo llama **frecuencia del término** (*term frequency*) y es notado como $\text{tf}_{t,d}$. A cada término del documento se le asigna el peso tf. El conjunto de pesos puede ser visto como un resumen cuantitativo del documento [58], donde el orden de las palabras no importan.

Si sólo se utiliza a la frecuencia de los términos como esquema de peso, a priori todos los términos son considerados igualmente importantes. Sin embargo, ciertas palabras tienen poco o ningún valor en la recuperación. Por esta razón, se introduce un mecanismo que refleje la importancia de los términos en la colección de documentos. Cuanto más ocurra un término en los documentos, menos discriminativo es este término entre los documentos, y consecuentemente menos útil en la recuperación [27, p. 241]. La **frecuencia inversa de documentos** (*inverse document frequency*) de un término se define como sigue:

$$\text{idf}_t = log\left(\frac{n}{\text{df}_t}\right),$$

donde n es la cantidad de documentos en la colección y df_t es la **frecuencia en documentos**, se refiere al número de documentos en la colección que tienen el término t. La base del logaritmo no es relevante para el ranqueo.

Como se ha indicado anteriormente, Spärck Jones [46] introdujo el peso idf como la *especificidad* de un término utilizando una interpretación heurística [7, p. 70]. En [26], se dio una justificación teórica de idf en términos del Modelo Probabilista Clásico.

Al combinar la definición de la frecuencia de un término y la frecuencia inversa de documentos se obtiene el *esquema de pesos* tf-idf,

$$\text{tf-idf}_{t,d} = \text{tf}_{t,d} \times \text{idf}_t\,. \tag{1.1}$$

Podemos decir que $\text{tf-idf}_{t,d}$ asigna al término t del documento d un peso que tiene las siguientes características [58, p. 109]:

- Mas alto, cuando t ocurre varias veces en un pequeño número de documentos (le brinda un gran poder de discriminación a esos documentos).
- Bajo, cuando el término ocurre pocas veces en el documento, o ocurre en muchos documentos (les ofrece una señal de relevancia menos pronunciada).

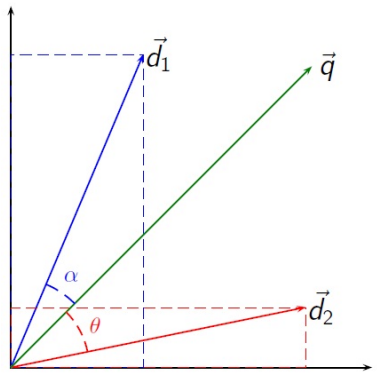


Figura 1.5: Similitud del coseno.

- Mas bajo, cuando el término ocurre virtualmente en todos los documentos.

Por lo tanto, se puede ver a cada documento como un vector cuyas componentes se corresponden a cada término en el vocabulario, con un peso para cada componente, como el la Ecuación 1.1. A lo largo de los años se han propuestos una gran cantidad de esquemas de pesos, muchos de los cuales son variaciones de tf-idf. En el esquema de pesos tf-idf, para aquellos términos del vocabulario que no ocurren en el documento su peso es cero.

A partir de la representación vectorial de los documentos, se puede calcular su relevancia ante una consulta mediante la similitud entre vectores.

La similitud del coseno

Se pueden definir a la consulta como un vector de la misma forma que un documento, es decir,

$$Q = (q_1, q_2, \ldots, q_k).$$

Es posible ranquear los documentos de acuerdo a la distancia entre los puntos que representa a un documento y la consulta. Se utiliza una *medida de similitud*, de modo tal que documentos con los puntajes más altos son los más similares a la consulta. La medida de similitud más utilizadas es la **similitud del coseno**, la cual calcula el coseno del ángulo entre el vector de la consulta y el documento, ver Figura 1.5. Se define de la siguiente manera:

$$score(D_i, Q) = coseno(D_i, Q) = \frac{D_i \cdot Q}{|D_i||Q|} = \frac{\sum_{j=1}^{k} d_{i,j}\ q_j}{\sqrt{\sum_{j=1}^{k} d_{i,j}^2\ \sum_{j=1} q_j^2}},$$

donde el numerador representa el producto interno de los vectores D_i y Q, y el denominador es el producto de su distancia euclidiana. Esta medida se puede utilizar para calcular la similitud entre documentos [58, p. 111], ya que es comparable a través de distintas consultas.

Computar la similitud del coseno entre el vector consulta y cada vector documento de la colección, ordenar los resultados, y tomar los mejores k documentos, puede resultar costoso. Es por esto que se utilizan índices invertidos, junto con una serie de heurísticas para mejorar los costos [58, p. 114]. Pese a que no existe una definición explícita de relevancia en el modelo espacio vectorial, se asume que la relevancia se relaciona con la

similitud del vector consulta con el documento. En otras palabras, documentos "cercanos" a la consulta son probablemente más relevantes [27, p. 242].

Las consultas sobre el espacio vectorial son fundamentalmente una forma de acumulación de evidencia [58, p. 136], donde la presencia de más términos de la consulta en un documento aumenta la puntuación del mismo. Por otra parte, la recuperación booleana requiere que el usuario especifique una fórmula para seleccionar los documentos a través de la presencia o ausencia de combinaciones específicas de palabras claves, sin inducir ningún orden entre los documentos recuperados. Se han propuesto modelos de recuperación que combinan a las consultas sobre el espacio vectorial y las consultas booleanas [88]. En la práctica, ambos modelos son utilizados en conjunto como se remarcó en la sección anterior. De aquí en adelante, nos referiremos al Modelo Espacio Vectorial con esquema de pesos tf-idf simplemente como TF-IDF. En la experimentación, se utiliza la implementación de la librería Lucene[7], la cual incluye *pivoted normalized document length* [31].

Puede encontrarse una justificación al buen rendimiento del Modelo Espacio Vectorial desde el modelo probabilístico BM25 que se muestra a continuación, ya que este último tiene por detrás argumentos teóricos que resultan en una fórmula que puede verse como un esquema de pesos tf-idf [82].

1.1.4. Okapi BM25

El sistema de ranking **BM25** es un modelo de recuperación probabilístico, desarrollado a partir de un principio que se formaliza a continuación.

Si se sume una noción de relevancia binaria, se puede definir una variable aleatoria R que tomará el valor 1 si el documento d es relevante para la consulta q y 0 de otra forma. Luego, es posible ordenar los documentos que se muestran al usuario estimando la probabilidad de relevancia con respecto a la necesidad de información, $P(R = 1|d, q)$. Robertson lo llamó **principio de ordenamiento por probabilidad** (*probability ranking principle*) [84].

Para ranquear los documentos con este principio se utiliza la razón de momios (una función creciente) de la siguiente manera:

$$O(R = 1|q, d) = \frac{P(R = 1|q, d)}{P(R = 0|q, d)} \propto \frac{P(q, d|R = 1)}{P(q, d|R = 0)}.$$

A partir de la distribución conjunta de documentos se tiene lo siguiente:

$$O(R = 1|q, d) \propto \frac{P(d|q, R = 1)}{P(d|q, R = 0)}.$$

Entonces, es necesario estimar $P(d|q, R = 1)$ y $P(d|q, R = 0)$, las probabilidades de los documentos relevantes e irrelevantes respectivamente. El principio no nos dice cómo estimarlas, pero dio origen a varios modelos que proponen distintas estimaciones.

Si $P(d|q, r)$ es estimado como una distribución múltiple de Bernoulli, entonces se asume independencia de los términos (también llamada hipótesis de naive bayes) y una representación binaria de los documentos, es decir se consideran a los documentos como vectores binarios, donde sólo se tiene en cuenta la presencia o ausencia de los términos en el documento. Este modelo es llamado Modelo Probabilista Clásico o también Modelo de Independencia Binaria (MIB) [85].

[7] `https://lucene.apache.org/core/8_0_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html`

En general, los modelos basados en el MIB no fueron efectivos, ya que si no se conocían algunos de los documentos relevantes de la consulta —llamados juicios de relevancia, ver Sección 6.1—, se obtenían fórmulas similares al peso clásico idf, no tenían una componente tf [26]. Esto hacía una gran diferencia en la efectividad del ranking, el cual caía cerca del 50 % si se ignoraba esta información [27, p. 249]. Por lo tanto, los modelos probabilísticos no consiguieron buenos rendimientos, hasta la llegada de BM25.

El esquema de pesos BM25

El ranking BM25 [81] consigue incluir en su formulación las frecuencia de los términos de los documentos y la consulta [27, p. 250]. Surge como una estimación de un modelo originado en el Principio de Ordenamiento de Probabilidad y el Modelo de Eliteidad[8], donde se asume que la frecuencia de los términos en los documentos, condicionados por su eliteidad, tiene una distribución de Poisson [83, p. 354].

Se define el puntaje de documento ante una consulta de la siguiente manera:

$$score(d,q) = \sum_{t \in q} \mathrm{idf}_t \,.\, \frac{(k_1+1)\,\mathrm{tf}_{t,d}}{k_1((1-b)+b\times(L_d/L_{ave}))+\mathrm{tf}_{t,d}}, \tag{1.2}$$

siendo L_d y L_{ave} la longitud del documento d y la longitud promedio en toda la colección, respectivamente. Los variables k_1 y b son parámetros de optimización del modelo, con valores por defecto en $k_1 \in [1.2, 2]$ y $b = 0.75$. Mientras que la frecuencia inversa de documentos se suele estimar de la siguiente manera:

$$\mathrm{idf}_t = log\left(\frac{N - \mathrm{df}_t + \frac{1}{2}}{\mathrm{df}_t + \frac{1}{2}} + 1\right),$$

donde N es la cantidad de documentos en la colección y $\mathrm{df_t}$ el número de documents que tienen al término t.

Estas definiciones son las más comunes, pero existe toda una familia de funciones de ranqueo BM25 que varían en sus componentes y parámetros, aunque sus rendimientos no difieren significativamente una de otra [47].

La implementación de este modelo no es muy distinta al modelo MEV, ya que al tratarse de otro esquema de pesos tf-idf, puede añadirse sin mayores complicaciones a un motor de búsqueda que utilice MEV. El esquema de pesos BM25 es utilizado ampliamente por su buen rendimiento a través de una gran variedad de colecciones [58, p. 215].

En el siguiente capítulo, continuaremos con otro modelo probabilístico que alcanza rendimientos similares a BM25 e incorpora la frecuencia de los términos directamente en el modelo —en lugar de agregarlo como una extensión para mejorar su rendimiento [27, p. 252]—, nos referimos a los Modelos del Lenguaje para la Recuperación de Información. Los modelos de recuperación vistos hasta ahora siguen un diseño heurístico, en cambio los Modelos del Lenguaje para la Recuperación de Información tienen un enfoque teórico [107]. El problema de determinar pesos de términos efectivos es reemplazado por la estimación de probabilidades, con una base formal [27, p. 260].

[8] Se hipotetiza que la ocurrencia de los términos en un documento tienen un elemento aleatorio o estocástico, que refleja una distinción oculta entre los documentos que tratan sobre el "concepto" representado por el término y aquellos que no [81]. Esta propiedad oculta se la llama de *eliteidad*. Si un término es de *elite* en un documento, entonces de alguna manera el documento trata sobre el concepto denotado por el término [83, p. 352].

Capítulo 2

Modelos del lenguaje para la RI

Los modelos del lenguaje se utilizan en muchas tareas del procesamiento del lenguaje natural, como reconocimiento del habla, traducción automática, etiquetado gramatical, recuperación de la información, entre otras aplicaciones. En lo que respecta a la recuperación de información, los modelos del lenguaje tienen un rendimiento comparable a BM25. Por su buen rendimiento empírico y potencial para añadir métodos estadísticos, han llamado la atención desde su primera publicación por Ponte y Croft [77].

Los modelos del lenguaje definen una distribución de probabilidades sobre las secuencias de palabras de un vocabulario. Se lo puede definir como una función que calcula la probabilidades de las cadenas de texto de algún alfabeto Σ, es decir [58, p. 219]:

$$\sum_{s \in \Sigma^*} P(s) = 1. \tag{2.1}$$

Para calcular la probabilidad de una secuencia de términos, podemos descomponer la probabilidad de una secuencia de eventos en la probabilidad de cada evento sucesivo condicionado con los eventos anteriores, mediante la regla de la cadena [58, p. 221]:

$$P(t_1 t_2 t_3 t_4) = P(t_1)P(t_2|t_1)P(t_3|t_1 t_2)P(t_4|t_1 t_2 t_3),$$

En la práctica, no podríamos estimar estas probabilidad a causa de la dispersión de los datos, la información disponible no suelen ser lo suficientemente grandes para estimar estas probabilidades adecuadamente. Por ello, se suele suponer la *independencia condicional y posicional*:

$$P_{uni}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2)P(t_3)P(t_4) = P_{uni}(t_4 t_2 t_1 t_3)$$

Este es un tipo especial de modelo de lenguaje llamado *unigrama*. En los unigramas el orden de las palabras es irrelevante (*bag og words*), cualquier orden de los términos tendrá la misma probabilidad.

Podemos definir modelos del lenguaje más complejos, como el modelo de lenguaje *bigrama*, el cual mantiene las condiciones sobre el término previo:

$$P_{bi}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2|t_1)P(t_3|t_2)P(t_4|t_3)$$

Ejemplo 2.1. Si la colección de documentos tiene cinco palabras diferentes, un posible modelo del lenguaje unigrama sobre dicha colección es $(0.2, 0.1, 0.35, 0.25, 0.1)$, donde cada número es la probabilidad de ocurrencia de la palabra. Si consideramos a un documento como una secuencia de palabras, entonces las probabilidades en el modelo del lenguaje predicen cuál será la próxima palabra de la secuencia. Si las cinco palabras de

este modelo del lenguaje son “niña”, “gato”, “el”, “niño” y “acaricia”, entonces las probabilidades predicen cuál de estas palabras será la próxima. Por ejemplo, la probabilidad de “niña gato” es la misma de “niña acaricia” (0.2×0.1) [27, p. 253]. De esta manera, se tiene un modelo para generar texto de acuerdo a su distribución.

Los modelos más complejos que los unigramas son útiles en tareas como el reconocimiento de voz, la traducción automática, corrección ortográfica, donde son necesarias las probabilidades de términos condicionados por su contexto [58, p. 221]. En la recuperación de información, la mayoría de los modelos son unigramas, ya que los modelos de lenguaje generalmente son estimados sobre un solo documento, por lo tanto, con pocos datos un modelo simple tiende a un mejor rendimiento [107, p. 158].

En siguiente sección, se detalla cómo utilizar a los modelos del lenguaje para ranquear documentos, más precisamente se describe el Modelo de Probabilidad de la Consulta (*query likelihood model*), a partir de siguiente idea: un documento será una buena respuesta ante una consulta si es probable que el modelo del lenguaje del documento genere la consulta, y esto sucederá si el documento contiene de manera frecuente las palabras de la consulta. Los modelos de recuperación que se detallan en las siguientes secciones, son la base del sistema de búsqueda propuesto.

2.1. Query likelihood model

En las aplicaciones de búsqueda, los modelos del lenguaje son utilizados para representar el tópico contenido en un documento [27, p. 255]. Por ejemplo, en el modelo del lenguaje de un documento sobre el fútbol argentino, deberíamos encontrar con altas probabilidades palabras asociadas con el fútbol y la Argentina . Seguidamente se describe el Modelo de Probabilidad de la Consulta, basado en la probabilidad de generar el texto de la consulta desde el modelos del lenguaje de los documentos.

Es posible ranquear los documentos a partir de la probabilidad $P(d|q)$, la cual es interpretada como la probabilidad de que el documento d sea relevante para la consulta q. Por la regla de Bayes, se tiene la siguiente fórmula:

$$P(d|q) = \frac{P(q|d)P(d)}{P(q)} \propto P(q|d)P(d). \tag{2.2}$$

La probabilidad a priori de la consulta $P(q)$ es la misma para todos los documentos, entonces a fines del ranqueo de documentos podemos ignorarla. Mientras que la probabilidad a priori del documento $P(d)$ puede ser útil para introducir criterios a favor de ciertos documentos [107, p. 149]. Por lo general, se la considera uniforme sobre todos los documentos, es decir, se la descarta. Entonces, se tiene lo siguiente:

$$P(d|q) \propto P(q|d).$$

Luego, en el Modelo de Probabilidad de la Consulta se considera a $P(q|d)$ como $P(q|\theta_d)$ [77]. Siendo $P(q|\theta_d)$ la probabilidad que tiene el texto q de acuerdo a la distribución dada por el modelo del lenguaje θ_d, esta se calcula de acuerdo al modelo de lenguaje unigrama, con una distribución multinomial de palabras [66, 42]. También, el modelo unigrama puede describirse como una distribución múltiple de Bernuolli [77], pero esta definición es menos utilizada que la distribución multinomial.

Por lo tanto, en el Modelo de Probabilidad de la Consulta los documentos son ranquedos por la probabilidad de que la consulta sea generada por el modelo del lenguaje del documento, de la siguiente manera [58, p. 224]:

Definición 2.1. (**query likelihood model, QL**) Sea q una consulta, d un documento y θ_d un modelo de lenguaje unigrama basado en el documento d. Se define el puntaje de un documento d con respecto a una consulta q como la probabilidad condicional $P(q|\theta_d)$,

$$score(q,d) = P(q|\theta_d) = \prod_{t \in q} P(t|\theta_d)^{c(t,d)}, \tag{2.3}$$

donde, $c(t,d)$ es la frecuencia del término t en el documento d.

Los pasos que deben seguirse para aplicar este modelo son:

- Inferir un modelo de lenguaje para cada documento.
- Estimar la probabilidad de generar la consulta de acuerdo a cada uno de estos modelos del lenguaje.
- Ordenar los documentos de acuerdo a estas probabilidades.

El modelo del lenguaje θ_d se utiliza para aproximar los tópicos que el autor del documento pensaba cuando lo escribía. Entonces, la probabilidad de generación de la consulta es medida como cuán probable es que el documento trate sobre el mismo tema de la consulta [27, p. 254].

Se espera que el usuario tenga un documento prototipo en mente y genere consultas basadas en palabras que aparecen en este documento. A veces, los usuarios tiene una idea razonable de los términos que son probables que ocurran en los documentos que son de su interés, y elijan términos en su consulta que distinguirán aquellos documentos de otros en la colección [58, p. 224]. A continuación, se muestran los métodos para estimar θ_d en base a al documento d.

2.1.1. Estimación de los modelos del lenguaje

Para calcular $P(q|\theta_d)$, es necesario estimar a los modelos del lenguaje definidos para cada documento. Se suele utilizar la *estimación por máxima verosimilitud* (*maximum likelihood estimation*, MLE). Este método permite estimar los parámetros de una distribución continua o discreta a partir de una muestra aleatoria, es decir, se asume observaciones independientes e idénticamente distribuidas (i.i.d *independent and identically distributed*). Encuentra los valores de los parámetros que maximizan la probabilidad de obtener los resultados de la muestra.

Sea un término t del vocabulario, el modelo de lenguaje unigrama de un documento d tiene el siguiente valor MLE [111]:

$$P_{mle}(t|\theta_d) = \frac{c(t,d)}{|d|}, \tag{2.4}$$

donde, $c(t,d)$ es la frecuencia del término t en el documento d, y $|d|$ es la longitud del documento (número total de términos).

Un problema generado por la dispersión de los términos en los documentos es que aquellos términos que no aparecen en el documento tienen probabilidad nula, esto da como resultado que todas las consultas que tengan un término que no está en el documento tengan cero probabilidad $P(q|\theta_d)$. Es decir, los documentos darán a una consulta una probabilidad no nula si todos los términos de la consulta aparecen en el documento. Pero, palabras asociadas al tópico del documento deberían tener una probabilidad de ocurrir incluso si no aparecen en el documento. Por ejemplo, un modelo del lenguaje

representando un documento acerca de videojuegos debería tener una probabilidad no nula para la palabra "RPG", incluso si dicha palabra no aparece en el documento [27, p. 255].

Entonces, es necesario *suavizar* las estimaciones de los modelos del lenguaje de documentos, es decir, reducir las probabilidades vistas en el documento y darle masa de probabilidad a términos no vistos. Un método de suavizado (*smoothing*) que funciona bien en la práctica consiste en interpolar la estimación del modelo de lenguaje del documento con un modelo de lenguaje estimado sobre toda la colección:

$$P(t|\theta_d) = \lambda P_{mle}(t|\theta_d) + (1-\lambda)P_{mle}(t|\theta_c), \tag{2.5}$$

donde $0 < \lambda < 1$ es un parámetro del suavizado y θ_c es el modelo del lenguaje construido a partir de toda la colección. Esta estimación recibe el nombre de **suavizado de Jelinek-Mercer**, o también llamada **interpolación lineal**.

Una segunda técnica de suavizado es el **suavizado de Dirichlet**,

$$P(t|\theta_d) = \frac{c(t,d) + \mu P_{mle}(t|\theta_c)}{|d| + \mu}. \tag{2.6}$$

En general, un valor de μ entre 1500 y 2500 es un buen valor por defecto.

Para consultas largas el suavizado de Jelinek-Mercer tiene un mejor rendimiento que el suavizado de Dirichlet. Mientras que para consultas cortas, el suavizado de Dirichlet se comporta mejor [109]. En definitiva, la elección del suavizado junto a sus parámetros tiene una gran influencia en el rendimiento del ranqueo.

Si retomamos la Definición 2.1 manteniendo $P(d)$, entonces el ranqueo para una consulta q con modelos del lenguaje para IR y suavizado de Jelinek-Mercer, se reduce de la siguiente manera:

$$P(d|q) \propto P(d) \prod_{t \in q} ((1-\lambda)P_{mle}(t|\theta_c) + \lambda P_{mle}(t|\theta_d))^{c(t,d)}.$$

Esta última fórmula está expresada como producto de probabilidades, pero en la práctica es mejor trabajar con sumas de logaritmos de probabilidades para evitar problemas de *underflow*. Entonces, se tiene:

$$\begin{aligned} P(d|q) &\propto log(P(d)) + \sum_{t \in q} c(t,d)\ log(\lambda P_{mle}(t|\theta_d) + (1-\lambda)P_{mle}(t|\theta_c)) \\ &= log(P(d)) + \sum_{t \in q} c(t,d)\ log\left(\lambda\ \frac{c(t,d)}{|d|} + (1-\lambda)\ \frac{c(t,c)}{|c|}\right), \end{aligned}$$

donde $c(t,c)$ es la frecuencia del término t en toda la colección de documentos, y $|c|$ es el número total de documentos en la colección.

En conclusión, el Modelo de Probabilidad de la Consulta puede definirse de diferentes maneras variando (1) θ_d (mediante bernoulli o multinomial), (2) métodos de estimación de θ_d (diferentes métodos de suavizado), o (3) con la probabilidad a priori del documento $P(d)$. Otra variación consiste en la relajación de algunos supuestos de independencia hechos por el modelo, de modo tal de capturar ciertas dependencias, como por ejemplo modelos de lenguaje bigramas [107].

Se ha demostrado que las fórmulas de ranqueo que utilizan suavizado de Jelinek-Mercer y Dirichlet pueden reescribirse como un esquema de peso TF-IDF [109] por lo tanto, computacionalmente el modelo es tan eficiente como cualquier método tradicional de TF-IDF que utilice índice invertido.

El Modelo de Probabilidad de la Consulta es un modelo simple que incorpora directamente la frecuencia de los términos. Este modelo tiene un rendimiento similar a BM25 cuando se utiliza el suavizado de Dirichlet [27, p. 260].

2.1.2. Extensiones a query likelihood model

Se presentan algunas extensiones al Modelo de Probabilidad de la Consulta. En primer lugar, se muestra la divergencia de Kullback-Leibler y luego se describen modelos de recuperación que consideran el tratamiento de documentos semiestructurados.

Modelo de recuperación con divergencia de Kullback–Leibler

Una de las limitaciones del modelo QL es la dificultad de incorporar un método de expansión de la consulta, dado que no se sabe cómo interpretar a una consulta modificada en este modelo. Por esta razón, se desarrollaron una familia de modelos probabilistas que utilizan la *divergencia de Kullback–Leibler* (KL) para la recuperación de información, el cual permite la expansión de la consulta.

En la divergencia de Kullback–Leibler, se definen dos modelos del lenguaje, uno para la consulta (θ_q) y otro para el documento (θ_d). Se asume que la consulta se observa como una muestra de un modelo del lenguaje de la consulta θ_q, el cual representa la necesidad de información del usuario, mientras que el documento es una muestra de un modelo del lenguaje del documento θ_d, esta representa el tema del documento. La divergencia de Kullback–Leibler[1] puede interpretarse como una "distancia", la cual mide cuán cerca están los ambos modelos del lenguaje y se ranquean a los documentos de acuerdo a esta medida, de manera similar al Modelo Espacio Vectorial.

El puntaje de un documento d con respecto a una consulta q, de acuerdo a la divergencia de Kullback–Leibler, es el siguiente:

$$\begin{aligned} score(d,q) &= -KL(\theta_q \parallel \theta_d) = -\sum_{w \in V} P(w|\theta_q)\ log\ \frac{P(w|\theta_q)}{P(w|\theta_d)} \\ &= \sum_{w \in V} P(w|\theta_q)\ log\ P(w|\theta_d) - \sum_{w \in V} P(w|\theta_q)\ log\ P(w|\theta_q) \\ &\propto \sum_{w \in V} P(w|\theta_q)\ log\ P(w|\theta_d), \end{aligned} \tag{2.7}$$

donde V es el vocabulario. El término $\sum_{w \in V} p(w|\theta_q) log P(w|\theta_q)$ (la entropía de la consulta) se ignora, ya que no influye en el ranking.

Solo resta estimar θ_q y θ_d. La estimación del modelo del lenguaje del documento θ_d puede ser la misma utilizada en el modelo de probabilidad de la consulta, ver Ecuación 2.4. Mientras que, el Modelo del Lenguaje de la Consulta θ_q, ofrece la oportunidad de mejorar su estimación utilizando modelos de expansión de la consulta, como se muestra en la Sección 2.2.

Además, puede verse que el modelo QL es un caso especial del modelo de divergencia KL, si se estima θ_q solo con la información de q [51], es decir:

$$P(w|\theta_q) = \frac{c(w,q)}{|q|}.$$

[1]La divergencia de Kullback-Leibler es una medida de divergencia asimétrica originaria de la teoría de información. Pese a no ser una verdadera distancia entre distribuciones, por no ser simétrica, es útil pensarla como una distancia [25].

La divergencia KL no incurre en un alto costo computacional, ya que es posible reescribir la fórmula de manera más eficiente que la Ecuación 2.7, considerando una suma de términos con probabilidades no nulas en θ_q y θ_d [71].

Modelos de lenguajes mixtos: representación de la información semiestructurada

La mayoría de los modelos de recuperación ignoran cualquier posible estructura de los documentos, nos referimos a los *campos* de un documento —p. ej. título, texto— . Es por esto que se han desarrollados adaptaciones para incluir la estructura de los documentos en los modelos de recuperación que se han mostrado en esta capítulo, por ejemplo modelos de espacio vectorial para XML [60], BM25F [106], lo cual potenciaría el rendimiento de las búsquedas.

Se presenta un modelo de recuperación que extiende al modelo QL, se lo llama Modelo del Lenguaje Mixto (*mixture language model*, MLM) o también Modelo Jerárquico del Lenguaje, permite ranquear a los documentos conociendo su estructura.

Sea $\mathcal{F}$ el conjunto de campos o partes de un documento, se anota con f_d al campo $f \in \mathcal{F}$ de un documento d. El modelo MLM combina los campos de un documento de acuerdo a sus pesos asociados, es decir:

$$P(t|\theta_d) = \sum_{f \in \mathcal{F}} \alpha_f P(t|\theta_{f_d}), \tag{2.8}$$

donde α_f es el peso o la importancia del campo f, tal que $\sum_{f \in \mathcal{F}} \alpha_f = 1$, y θ_{f_d} es el modelo del lenguaje del campo f en el documento d. Este valor puede reflejar un conocimiento a priori del dominio, o se puede calcular a partir de un conjunto de entrenamiento. El origen de esta fórmula se puede ver de la siguiente manera:

$$\begin{aligned} P(t|d) &= \sum_{f \in \mathcal{F}} P(t|f_d) P(f_d|d) \\ &= \sum_{f \in \mathcal{F}} \alpha_f P(t|f_d) \\ &= \sum_{f \in \mathcal{F}} \alpha_f P(t|\theta_{f_d}), \end{aligned} \tag{2.9}$$

donde $\alpha_f = P(f_d|d)$.

Por lo tanto, el ranking de un documento siguiendo el modelo QL es el siguiente:

$$\begin{aligned} P(q|d) &= \prod_{i=1}^{m} P(q_i|\theta_d) \\ &= \prod_{i=1}^{m} \sum_{f \in \mathcal{F}} \alpha_f P(q_i|\theta_{f_d}), \end{aligned} \tag{2.10}$$

donde $q = \langle q_1, \ldots, q_m \rangle$.

La generación de la consulta por parte de un documento semiestructurado es el resultado de dos pasos: la selección del campo f de un documento semiestructurado d de acuerdo a la probabilidad α_f, y la generación de la consulta a partir del campo seleccionado f_d [107, p. 183]. Por último, el valor de $P(q_i|\theta_d)$ se estima con los métodos vistos anteriormente, por ejemplo podemos utilizar el suavizado de Jelinek-Mercer, ver Ecuación 2.5, reemplazando d por f_d.

Modelo de recuperación probabilístico para datos semiestructurados

Una variante al modelo anterior desarrollada por Kim et al. [50], denominada Modelo de Recuperación Probabilístico para Datos Semiestructurados (*Probabilistic Retrieval Model for Semistructured Data*, PRMS) propone una forma no supervisado de estimar los pesos de los campos de los documentos. Además, en lugar de utilizar un peso estático para todos los términos, los pesos varían para cada término. Para esto, los pesos se definen a partir de una distribución de los términos sobre los respectivos campos.

El peso α_f, en la Ecuación 2.8, es reemplazado con la probabilidad de *mapear* el término t al campo f, $P(f|t)$, es decir:

$$P(t|\theta_d) = \sum_{f \in \mathcal{F}} P(f|t) P(t|\theta_{f_d}), \tag{2.11}$$

donde $P(f|t)$ se define, por la regla de Bayes y el teorema de la probabilidad total, de la siguiente manera:

$$P(f|t) = \frac{P(t|f)P(f)}{P(t)} = \frac{P(t|f)P(f)}{\sum_{f' \in \mathcal{F}} P(t|f')P(f')}.$$

La probabilidad $P(f)$ puede incorporar conocimiento a priori o dejarse uniforme. Mientras que, la probabilidad $P(t|f)$ es estimada usando el modelo de lenguaje de toda la colección en dicho campo, es decir $P(t|f) \cong P(t|\theta_{f_c})$. Este modelo asume una colección donde los campos pueden caracterizarse por distribuciones de términos distintivas y sólo funciona si se cumple dicha condición.

2.2. Expansión de consultas

El gran problema que tienen los sistemas de búsqueda es la *discordancia de términos* (*vocabulary mismatch problem*), esto sucede cuando las palabras que utiliza el usuario no coinciden con los términos de los documentos que son relevantes, esto puede deberse a la polisemia (la misma palabra con diferentes significados, por ejemplo "banco") y la sinonimia (diferentes palabras con el mismo o similar significado, por ejemplo "televisión" y "tv"). Los sinónimos junto con las inflexiones de las palabras (como la forma plural, por ejemplo "televisión" y "televisores") puede resultar en no retornar documentos relevantes, con una reducción de la exhaustividad (la habilidad del sistema de retornar todos los documentos relevantes). Mientras que, la polisemia puede arrojar documentos irrelevantes o erróneos, esto implica un decrecimiento de la precisión (la habilidad del sistema de retornar solo documentos relevantes) [21, p. 2]. Además, para los usuarios puede ser difícil formular consultas bien diseñadas para la recuperación, sin un conocimiento preciso de la colección de documentos. Estas dificultades motivaron a las reformulaciones de la consulta, es decir, añadir a la consulta original otras palabras que capturen la intención del usuario o simplemente produzcan una consulta más útil. Este proceso de modificación de la consulta es denominado en la literatura como **expansión de la consulta**.

Seguidamente, se describe en términos generales el mecanismo de la expansión de la consulta y sus etapas, tomando como referencia a [21]. Luego, se explica el método empleado en el sistema propuesto, la retroalimentación por (pseudo) relevancia.

Todos los modelos de recuperación presentados anteriormente (TF-IDF, BM25, QL) computan la importancia que tienen los términos que aparecen en la consulta y en los documentos para generar los resultados. Es posible expresar la función de ranking entre una consulta q y un documento d de estos modelos como sigue:

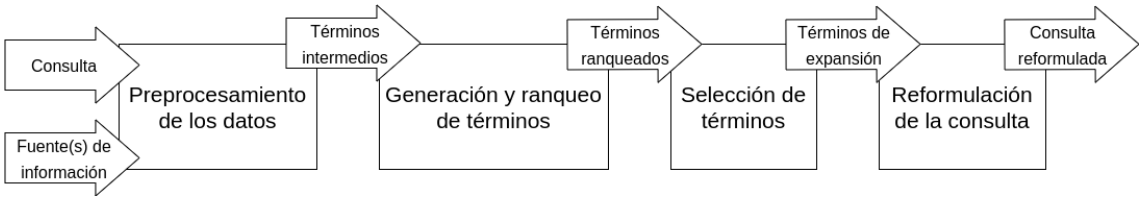


Figura 2.1: Etapas de la expansión de una consulta.

$$score(d, q) = \sum_{t \in q \cap d} w_{t,q} \cdot w_{t,d},$$

siendo $w_{t,q}$ y $w_{t,d}$ los pesos del término t en la consulta q y el documento d, respectivamente, de acuerdo al esquema de pesos del sistema.

Un sistema de expansión de la consulta, utiliza como entrada a la consulta original del usuario junto con una fuente de información —la llamamos fuente de expansión de la consulta— para generar los términos que conformarán a la consulta expandida. Entonces, la salida del sistema será una nueva consulta q' formada por los términos de la expansión y sus pesos asociados w'. Estos pesos, en general, son utilizados para calcular el ranking final, es decir:

$$score(d, q') = \sum_{t \in q' \cap d} w'_{t,q'} \cdot w_{t,d}, \tag{2.12}$$

Como fuente de expansión de la consulta puede utilizarse a la misma colección de documentos del sistema, más adelante se abordan los tipos de fuentes de información. Por otra parte, se puede clasificar a la expansión de la consulta en dos categorías:

- **Expansión automática de la consulta**: el sistema automáticamente reformula la consulta sin ninguna intervención del usuario.
- **Expansión interactiva de la consulta**: el sistema sugiere reformulaciones de la consulta, la decisión la hace el usuario.

En los algoritmos de expansión de la consulta propuestos, Sección 5.3, se considera como expansión interactiva a la sugerencia de conceptos, en lugar de términos.

Las etapas que componen a la mayoría de los métodos de expansión de la consulta —tanto automáticos como interactivos— se pueden ver en la Figura 2.1, son los siguientes [5] [21]:

1. El **preprocesamiento de los datos** consiste en trasformar la información en un formato que pueda ser procesado eficientemente en las etapas posteriores. El método más sencillo consiste en utilizar los técnicas de normalización de texto (Sección 1.1.1), por ejemplo remoción de palabras vacías y *stemming*.

2. En la **generación y ranqueo de términos** se producen los términos candidatos para la expansión de la consulta, según al algoritmo de expansión de la consulta que se utilice en esta etapa. A cada término del vocabulario se les asigna un puntaje $score(t)$ que mide su utilidad para la expansión.

3. En la **selección de términos** se obtienen los términos finales que formarán parte de la expansión de la consulta. En la expansión automática, en general, se toman los primeros k términos, generados en la etapa anterior. En cambio, en la expansión interactiva, el usuario interviene para seleccionar los términos de la expansión.

4. La **reformulación de la consulta** genera la consulta expandida. En esta etapa suele utilizarse *repesado de la consulta*, donde se asignan nuevos pesos a la consulta original. Su formulación general es la siguiente. Sea q' la consulta expandida, q la consulta original y λ un parámetro para pesar la contribución de los términos de la consulta y los de la expansión. Luego,

$$w'_{t,q'} = (1 - \lambda)\ w_{t,q} + \lambda\ score(t),$$

siendo $score(t)$ el peso asignado al término de la expansión t por la etapa *generación y ranqueo de términos*. Por lo tanto, los nuevos pesos de la consulta expandida q' son una combinación lineal de sus pesos originales con los pesos de los términos de la expansión. Se pueden utilizar otros tipos de reformulaciones como generación de una consulta booleana [37].

Por último, la consulta expandida se emplea en el sistema de búsqueda para obtener los resultados finales como se muestra en la Ecuación 2.12. En la siguiente sección, se describe un método de expansión de la consulta, llamado retroalimentación por (pseudo) relevancia.

2.2.1. Retroalimentación por relevancia y retroalimentación por pseudo relevancia

Los modelos de expansión de la consulta utilizan una fuente de información de la cual se extraen los términos a utilizar en la expansión. Se pueden clasificar en dos grandes grupos, de acuerdo al tipo de fuente de información empleada, son (1) análisis global y (2) análisis local [5].

En el *análisis global*, las técnicas de expansión usan los términos de una base de conocimiento o de un gran corpus. Algunos de estos métodos son los siguientes:

- Expansión de la consulta con tesauros.
- Expansión de la consulta a partir de la generación automática de tesauros.

El *análisis local* incluye técnicas de expansión que toman los términos de expansión a partir de los documentos recuperados en respuesta a la consulta inicial del usuario (no modificada). Los métodos básicos son:

- La retroalimentación por relevancia.
- Retroalimentación por pseudo relevancia.

En esta tesina se emplea un método híbrido, ya que si bien utilizamos a los métodos locales nombrados anteriormente, lo hacemos sobre una fuente de información construida a partir de una base de conocimiento. Se brindan más detalles en la descripción del sistema de búsqueda propuesto, ver Capítulo 5.

La **retroalimentación por relevancia** (*relevance feedback*) depende de la interacción con el usuario para identificar los documentos relevantes de los resultados que se obtienen de la consulta inicial. El contenido de los documentos seleccionados es utilizado para ajustar los pesos de los términos de la consulta original y/o agregar nuevos términos a la consulta [21]. El procedimiento básico se muestra en la Figura 2.2 y se describe a continuación:

- El usuario ingresa una consulta (corta, simple), q.

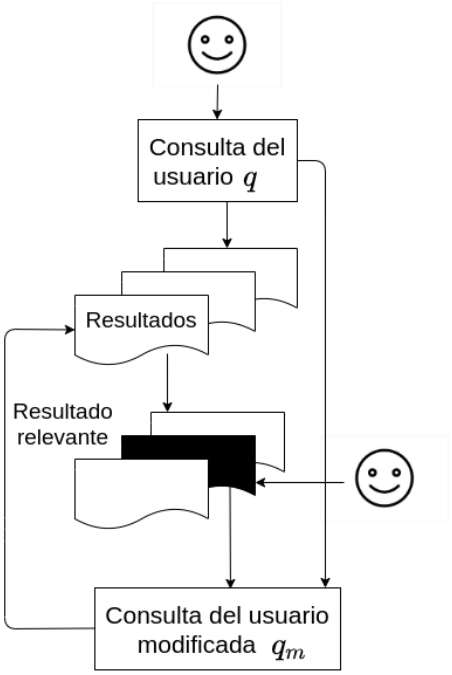


Figura 2.2: Retroalimentación por relevancia (Fuente [7, p. 179]).

- El sistema retorna un conjunto de resultados iniciales.
- El usuario marca algunos de los documentos recuperados como relevantes o no relevantes.
- El sistema computa una mejor representación de la necesidad de información basada en la retroalimentación con el usuario, generando una consulta expandida q_m.
- El sistema muestra un nuevo conjunto de resultados a partir de q_m.

El sistema de retroalimentación por relevancia puede hacer una o más iteraciones de estos últimos 3 pasos, y en ese caso se lo llama **retroalimentación por relevancia iterativa** (*iterative relevance feedback*) [16]. Este procedimiento explota la idea de que puede ser difícil formular una buena consulta cuando el usuario no conoce la colección, pero puede ser fácil juzgar documentos particulares, lo que le daría sentido a un refinamiento iterativo de la consulta de este tipo [58, 163].

Pedirle información al usuario puede resultar costoso, ya que tienden a ser reticentes a proveer este tipo información, por lo que se han desarrollados otros métodos automáticos que hacen uso de información potencialmente relevante para el usuario. Una de la maneras de evitar la intervención del usuario, se la llama **retroalimentación por pseudo relevancia** (*pseudo-relevance feedback*) o **retroalimentación ciega** (*blind relevance feedback*), en la que se asumen que los primeros k resultados de la consulta inicial son relevantes, para luego reformular la consulta.

Estos algoritmos son una de las técnicas más populares de expansión automática de la consulta. La evidencia sugiere que tienden a mejorar los resultados con respecto a los métodos de análisis global. Si bien los métodos de retroalimentación por relevancia y retroalimentación por pseudo relevancia han sido extensamente investigados en la literatura científica, y han mostrado ser efectivos para mejorar el ranqueo, en general

no han sido incorporados a los motores de búsquedas de la industria. En el caso de la retroalimentación por pseudo relevancia puede deberse principalmente a que el proceso de automatización resulta impredecible. Si el resultado no contiene muchos documentos relevantes, la expansión puede no resultar beneficiosa, e incluso para algunas consultas puede empeorar significativamente los resultados [27].

2.2.2. Modelo de relevancia

Es posible representar el tema de una consulta como un modelo del lenguaje, se lo puede llamar *modelo de relevancia* ya que representa el tema de los documentos relevantes. Se puede ver a la consulta como una pequeña muestra de texto generado por el modelo de relevancia, y a los documentos relevantes como muestras más grandes generadas por dicho modelo. Los documentos pueden ranquearse por la similaridad de su modelo del lenguaje con el modelo de relevancia, un documento cuyo modelo sea similar al modelo de relevancia es muy probable que sea del mismo tema [27, p. 261].

Si vemos a las palabras de la consulta como una muestra del modelo de relevancia R, entonces es razonable basar las probabilidades de nuevos términos de palabras en las palabras de la consulta ya vista. Más formalmente, se puede relacionar la probabilidad de una palabra w con la probabilidad condicionada de observarla dado los términos $q_1, \ldots, q_n$ ya observados:

$$P(w|R) \approx P(w|q_1, \ldots, q_n).$$

Lavrenko y Croft [54] presentan dos métodos para estimar esta probabilidad condicionada. La que obtiene el mejor rendimiento es llamada **RM1**, se asume que los términos en la consulta y en los documentos relevantes son muestras independientes e idénticamente distribuidas del modelo de relevancia (i.i.d):

$$P(w|R) \propto \sum_{\theta_d \in \Theta} P(\theta_d) P(w|\theta_d) \prod_{i=1}^{m} P(q_i|\theta_d), \tag{2.13}$$

donde Θ representa el conjunto de modelos del lenguaje suavizados de documentos. En general, se considera a $P(\theta_d)$ uniforme y es ignorada. Mientras que, la expresión $\prod_{i=1}^{m} P(q_i|\theta_d)$ es el puntaje del modelo QL (ver Ecuación 2.3) para el documento d.

Por lo tanto, $P(w, q_1, \ldots, q_n)$ es un promedio ponderado de las probabilidades de los modelos del lenguaje para w de un conjunto de documentos, donde los pesos son los puntajes del modelo QL para dichos documentos [27, p. 264]. Sólo importan los primeros documentos, usualmente los 10-50 documentos con $P(q|\theta_d)$ más alto, porque todos los demás tendrán valores muy bajos o nulos. Por razones similares, se considera un pequeño número (10-25) de términos para representar al modelo de relevancia, aquellos con las probabilidades más altas.

Como se dijo, hay poca diferencia entre el modelo de relevancia y el modelo de la consulta, ya que ambos modelan lo que le interesa al usuario. Es decir, se puede ver a $P(w|R)$ como $p(w|\theta_q)$ [107, p. 177]. Por lo tanto, el Modelo de Relevancia puede ser visto como un proceso de retroalimentación por pseudo relevancia, donde la consulta es expandida al agregarle términos de documentos que son similares a la consulta.

El modelo de recuperación que realiza la expansión del Modelo de Relevancia y utiliza de la divergencia KL, se anota como **RM3** y consiste en los siguientes pasos:

1. Ranquear los documentos utilizando el modelo QL.

2. Seleccionar los primeros documentos.

3. Calcular la probabilidad del Modelo de Relevancia mediante la Ecuación 2.13, lo notamos como $P(w|\hat{\theta}_q)$.

4. Ranquear los documentos utilizando la divergencia de KL (Ecuación 2.7):

$$score(d, q) = \sum_{w \in V} P(w|\hat{\theta}_q)\ log\ P(w|\theta_d).$$

En general, cuando se estima el Modelo de Relevancia se lo combina con el modelo del lenguaje original de la consulta mediante una interpolación, esto evita que la consulta cambie demasiado con respecto a la intensión original del usuario, este es un problema conocido como **desvio de la consulta** (*query drift*):

$$P(w|\hat{\theta}_q) = (1 - \lambda_q)P(w|\theta_q) + \lambda_q P(w|R), \tag{2.14}$$

siendo λ_q un parámetro que controla la influencia del modelo expandido de la consulta, $P(w|R)$ es la estimación del modelo de relevancia de la Ecuación 2.13 y $P(w|\theta_q)$ la estimación MLE del modelo del lenguaje de la consulta inicial, es decir,

$$P(w|\theta_q) = \frac{c(w, q)}{l_q},$$

donde $c(w, q)$ es el número de veces que aparece w en q y l_q es la cantidad total de términos en q.

La Ecuación 2.14 deja en claro que el Modelo de Relevancia es un proceso de expansión de la consulta [27, p. 265]. RM3 es considerado el modelo del estado del arte para la retroalimentación por pseudo relevancia.

En la siguiente sección, se muestra una modificación al Modelo de Relevancia para ser un mecanismo de "verdadera" retroalimentación por relevancia.

2.2.3. Modelo iterativo de relevancia

Existen tres tipos de métodos clásicos de retroalimentación por relevancia, aquellos basados en el Modelo Espacio Vectorial, en el Modelo Probabilista Clásico y en los Modelos del Lenguaje para la Recuperación de Información. El primer modelo de retroalimentación por relevancia se basó en el espacio vectorial, se lo llamó Rocchio [86]. Este modelo refine al vector de la consulta más cerca del centro (centroide) de los documentos relevantes y más lejos de los documentos no relevantes. Por otro lado, el Modelo Probabilista Clásico incorpora la retroalimentación por relevancia desde su primera publicación [85] para la estimación de los parámetro del modelo. Los términos de expansión de la consulta se calculan sobre la probabilidad de sus ocurrencias en los documentos relevantes y en los documentos no relevantes. En el caso de los modelos del lenguajes para RI podemos encontrar varios modelos con retroalimentación por pseudo relevancia, como el Modelo de Relevancia visto en la sección anteriormente, otro modelo es el *Simple Mixture Model*[2] [108].

La mayoría de los estudios se enfocaron en el desarrollo de métodos de retroalimentación por relevancia efectivos que mejoraran la recuperación del sistema en una sola iteración, los usuarios proveían de documentos relevantes sobre los primeros 10 o más resultados iniciales [87]. Se desarrollaron pocos modelos de retroalimentación por relevancia iterativa, el primero fue propuesto por Aalsberg et al., se basó en Rocchio. En su

[2]Se define al modelo de la consulta expandida como una interporlación del modelo de la consulta original y un modelo de los temas en los documentos relevantes (*feedback topic model*), el cual es estimado con el algoritmo de *machine learning* Expectation-Maximization.

trabajo, se les pidió a los usuarios que juzgaran el resultado de cada iteración, de modo tal que la consulta se refinara iterativamente con la retroalimentación. Este enfoque mostró mejores rendimientos en comparación al método con relevancia de una sola iteración [16].

Originalmente el Modelo de Relevancia se propuso como un método de retroalimentación por pseudo relevancia, pero también se han propuesto versiones con retroalimentación por relevancia, es decir que el usuario debe seleccionar aquellos documentos que considera relevantes. A continuación, se muestra una versión iterativa del Modelo de Relevancia, propuesta en [16].

Sea $R^{(i)}$ el conjunto de documentos juzgados como relevantes en la i-ésima iteración. Entonces, se define el conjunto de todos los documentos relevantes $RP^{(i)}$, de la siguiente manera:

$$RP^{(i)} = RP^{(i-1)} \cup R^{(i)}, \tag{2.15}$$

donde $i > 0$, $RP^{(0)} = \emptyset$.

Luego, el modelo de relevancia en la i-ésima iteración ($i > 0$) se puede estimar como sigue [53, p. 70],

$$P^{(i)}(w|R) = \frac{1}{|RP^{(i)}|} \sum_{d \in RP^{(i)}} P(w|\theta_d). \tag{2.16}$$

Si $RP^{(i)} = 0$, consideramos $P^{(i)}(w|R) = 0$.

Luego, el modelo de lenguaje de la consulta en la i-ésima iteración es la combinación lineal con el modelo de la consulta original, como se muestra a continuación:

$$P^{(i)}(w|\theta_q) = \begin{cases} (1-\lambda_q)P^{(0)}(w|\theta_q) + \lambda_q P^{(i)}(w|R) & si \;\; i > 0 \\ \frac{c(w,q)}{l_q} & sino \end{cases} \tag{2.17}$$

Por último, los documentos son ranqueados con la divergencia de KL, es decir:

$$score^{(i)}(d,q) = \sum_w P^{(i)}(w|\theta_q) \; log \; P(w|\theta_d). \tag{2.18}$$

La retroalimentación iterativa por relevancia mejora el rendimiento de la retroalimentación con una sola iteración. En los modelos iterativos se pueden identificar resultados relevantes con pocas iteraciones gracias a un re-ranqueo más "temprano" en comparación a la retroalimentación por relevancia con una sola iteración. Pero, habría menos resultados disponibles en las primeras iteraciones que aprovecharían los juicios de relevancia [16]. Estos mecanismos de retroalimentación iterativa necesitan suficientes documentos relevantes para estimar un modelo de la consulta robusto, para evitar el problema del desvio de la consulta, ya que si se tienen pocos documentos relevantes harían a la consulta más vulnerable de los temas no relevantes contenidos en los documentos juzgados como relevantes [16].

Hasta aquí, se mostraron los conceptos básicos para la recuperación de documentos basados en modelos del lenguajes junto a mecanismos de expansión de la consulta. En el siguiente capítulo, se describen algoritmos de búsqueda de entidades, pertenecientes a una base de conocimiento, más adelante, en el Capitulo 4 se detalla a este tipo de fuente de información. Las búsqueda de entidades permiten mejorar la recuperación de documentos, dando lugar a un sistema de búsqueda semántico.

Capítulo 3

Búsqueda orientada a entidades

La recuperación de entidades tiene como objetivo responder a consultas mediante una lista de ranqueada de entidades, ya que ciertas necesidades de información pueden responderse mejor mediante entidades, en lugar de retornar documentos, por ejemplo "países limítrofes de Argentina", "ganadores del quini 6", "creadores del himno nacional argentino".

Las *entidades*[1] son objetos unívocamente identificables, con nombre/s, atributos y relaciones con otras entidades, por ejemplo, personas, localizaciones y organizaciones. La búsqueda orientada a entidades puede verse como un paradigma de acceso y organización de la información centrada en las entidades, sus atributos y relaciones. Se la define de la siguiente manera [9, p. 58]:

Definición 3.1. Dada una consulta de palabras claves q y un catálogo de entidades $\mathcal{E}$, la recuperación de entidades consiste en retornar una lista de entidades $\langle e_1, \ldots, e_k \rangle$, con $e_i \in \mathcal{E}$, de acuerdo a la relevancia de cada entidad con respecto a q. Esta relevancia se infiere de acuerdo a una colección no estructurada o semi-estructurada de datos.

Se considera la existencia de un *catálogo de entidades*, este es un diccionario con los nombres de las entidades junto a sus identificadores. Este catálogo puede ser una base de conocimiento, como una ontología. Mientras que la colección de datos puede ser una colección de documentos, donde cada documento representa a una entidad del catálogo. Es decir, para cada entidad del catálogo se crea un "documento" —llamado *descripción o perfil de la entidad*—, el cual mantiene toda la información que se conoce acerca de la entidad que representa. Este enfoque propone ranquear a las entidades utilizando los algoritmos de recuperación de documentos tradicionales, a partir de las descripciones de las entidades. De esta manera, se consigue utilizar sin modificaciones, los modelos diseñados para ranquear documentos.

En la Figura 3.1 se muestra la arquitectura de un sistema de búsqueda orientado a entidades. Puede verse que la única diferencia con respecto a un sistema de recuperación convencional es la inclusión de un repositorio de conocimiento, el cual contiene la información de la entidades, debería contener, al menos, el catálogo de las entidades.

En particular, en esta tesina se emplea un sistema de búsqueda de entidades, donde las entidades son los individuos de una ontología legal, esto forma parte del modelo de búsqueda de documentos propuesto (Sección 5.3).

En este capítulo se presentan ideas básicas de la recuperación de entidades, se explica la construcción de las representaciones de las entidades a partir de una base de conocimiento (Sección 3.1), los sistema de ranking de entidades (Sección 3.2) y una serie

[1] En la literatura, también se utiliza el término "concepto" en lugar de entidad.

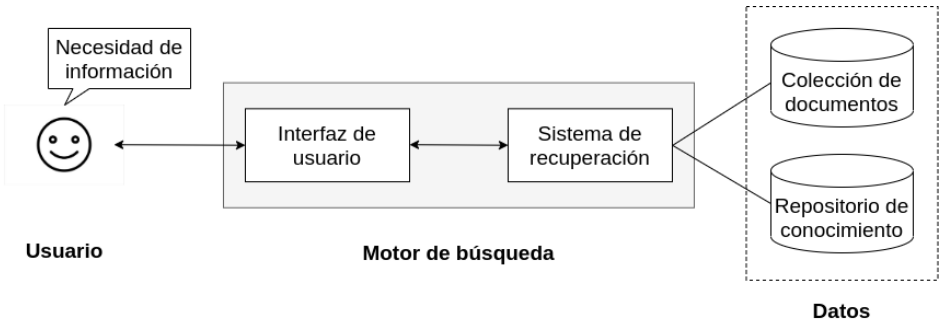


Figura 3.1: Arquitectura de un sistema de búsqueda orientada a entidades (Fuente [9, p. 12]).

de expansiones a los métodos de ranking de entidades básicos(Sección 3.3), por último, se muestra un modelo de expansión de la consulta basado en entidades, es decir que incorpora la información de las entidades (Sección 3.4).

3.1. Representación de las entidades

La estrategia para la recuperación de entidades descripta anteriormente [10, 11, 13], se llama Modelo Centrado en Candidatos (*candidate-centric models*), esto es construir una representación textual de cada entidad y ranquearla ante una consulta de acuerdo a modelos de recuperación tradicionales. Una segunda estrategia, denominada Modelo Centrado en Documentos (*document-centric models*), comienza hallando los documentos relevantes a la consulta y luego se identifican a las entidades asociadas a estos documentos, es decir que las entidades no son modeladas directamente. Este último método suele ser el preferido, ya que ha mostrado ser más robusto y efectivo que el modelo centrado en candidatos [80], además puede implementarse fácilmente, no requiere la construcción de un índice dedicado a almacenar la información de las representaciones textuales de las entidades. Sin embargo, un Modelo Centrado en Documentos no contempla otras fuentes de datos que no sean los documentos de la colección, es decir que, no puede ser utilizarse una base de conocimiento. Por ello, en esta sección se describe la construcción de las representaciones de las entidades en un Modelo Centrado en Candidatos, el cual sí admite a una base de conocimiento como fuente de información del sistema.

La construcción de una representación textual para cada entidad del catálogo que incluya todo lo que se conoce acerca de la misma, puede realizarse de diferentes maneras de acuerdo al tipo de fuente de información que se utilice (no estructurada o semi-estructurada [9, p.]). A continuación, se detalla un método para la construcción de las descripciones de entidades definidas en una base de conocimiento en formato RDF, ya que es el tipo de datos con el que se trabajará en esta tesina.

En el modelo de datos RDF (ver Capítulo 4) puede verse a un conjunto de tripletas sujeto-predicado-objeto (SPO) como un grafo dirigido, donde cada arista, el predicado, es etiquetada con una URI (Uniform Resource Identifier) y cada nodo con un literal o URI. En la construcción de la descripción de una entidad es de interés su vecindad en el grafo de la base de conocimiento, en la Figura 3.2 puede verse un ejemplo.

El objetivo consistirá en concatenar todo el texto que se encuentre en las tripletas SPO donde ocurra la entidad de interés. En un principio, se podría representar a cada predicado como un campo de la descripción de la entidad. Pero, en grandes bases de conocimientos el número de predicados es del orden de miles, aunque la mayoría de la

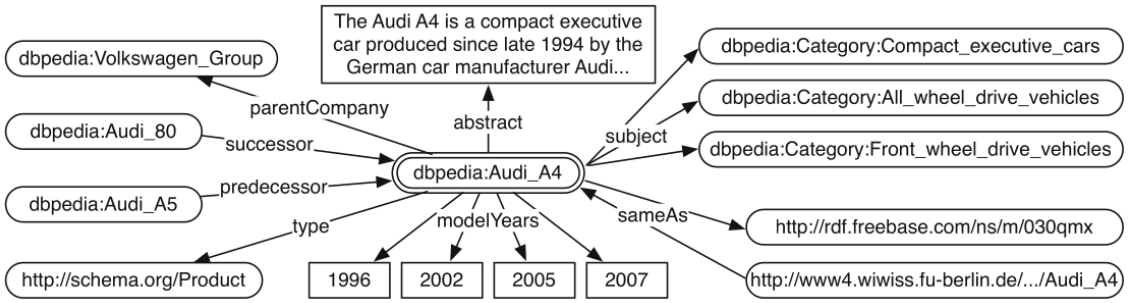


Figura 3.2: Extracto de un grafo RDF. Los rectángulos redondeados representan entidades (URIs) y los rectángulos representan valores literales. (Fuente [69])

Pred. type	**Value**
Name	Audi A4
Attributes	1996 2002 2005 2007 The Audi A4 is a compact executive car...
OutRelations	Volkswagen Group Compact executive cars All wheel drive vehicles Front wheel drive vehicles Product Audi A4
InRelations	Audi 80 Audi A5 Audi A4

Figura 3.3: Descripción de la entidad "Audi A4", Figura 3.2 (Fuente [69]).

entidades tiene un número pequeño de predicados asociados a ellas. Por lo tanto, en estos casos, la estimación de los pesos de miles de campos en el modelo de ranqueo es computacionalmente prohibitivo.

Por esta razón, se utiliza un método denominado *predicate folding* [9, p. 71], en el que se agrupan a los predicados en pequeños conjuntos de categorías. De esta manera, se tiene un solo campo en las descripciones de entidades por cada una de estas categorías, lo que resulta en un número de pesos muchos más manejable.

Los predicados pueden agruparse basados en sus tipos [69] o en su importancia [17], asignada manualmente. La Figura 3.3 muestra la descripción la entidad que se muestra en la Figura 3.2. En este caso, los campos se definieron de acuerdo a los tipos de predicados, siendo e la entidad de interés y el sujeto de una tripleta SPO:

- *Name*: contiene el nombre de la entidad, se utiliza por ejemplo con los predicados `rdfs:label` y `foaf:name`.
- *Attributes*: incluye todos los objetos con valores literales, excepto aquellos incluidos en el campo *name*.
- *Outgoing relations*: contiene objetos URI apuntados por la entidad e. Tanto en este caso como en *incoming relations* la URI debe ser resulta (Sección 5.2).
- *Incoming relations*: tiene los sujetos que son URIs de todas las tripletas SPO donde e es el objeto de la relación.

Además, se suele utilizar el campo *catch-all* (o contenido), el campo que amalgama toda la información relacionada a la entidad. La configuración anotada como

"título+contenido" es considerada un buen punto de partida en el estado de arte [39] La importancia de otros campos puede variar según el tipo de consultas [110].

Se describe a continuación mas formalmente el método *predicate folding* [9, p. 72]. Sea una base de conocimiento $\mathcal{K}$ con tripletas SPO, $(s, p, o) \in \mathcal{K}$. Hay dos tipos de patrones de tripletas que contienen la información directamente relacionada a una entidad e: (1) relaciones de salida, donde la entidad es el sujeto de la tripleta, es decir (e, p, o), y (2) relaciones de entrada, donde la entidad es el objeto de la tripleta, es decir (s, p, e). En *predicate folding* se definen las asociaciones entre los campos de una descripción y los predicados de la base de conocimiento. Estas son las funciones $\mathcal{M}_{out}(p)$ y $\mathcal{M}_{in}(p)$ que retornan el conjunto de campos que se asocian al predicado p para las relaciones de entrada y salida respectivamente (devuelve $\emptyset$ si el predicado no es mapeado a ningún campo).

Luego, la secuencia de términos de un campo de una entidad es la siguiente:

$$f_{e_t} = \bigcup_{\substack{(e,p,o)\in\mathcal{K} \\ f\in\mathcal{M}_{out}(p)}} o_t \bigcup_{\substack{(s,p,e)\in\mathcal{K} \\ f\in\mathcal{M}_{in}(p)}} s_t,$$

donde $\bigcup$ es el operador de concatenación de caracteres[2]. La secuencia de términos del objeto y sujeto de una tripleta SPO se denota como o_t y s_t, respectivamente.

3.2. Ranking de entidades

En la sección anterior se vio como construir las representaciones de las entidades. Ahora, es posible utilizar a los modelos tradicionales de recuperación de la información para que dada una consulta se retorne una lista de entidades.

A continuación, se presenta el modelo *query likelihood* adaptado para la recuperación de entidades, los documentos pasan a referirse a las descripciones de las entidades, en las fórmulas esto se refleja reemplazando la variable d por e.

Modelos del lenguaje para la recuperación de entidades

Calcular $score(e, q)$, para cada entidad e del catálogo de entidades, puede realizarse con cualquier modelo de recuperación visto en la Capítulo 1 y 2. Como se obtuvieron buenos resultados utilizando a los modelos del lenguaje, se toma al modelo QL y sus variantes para puntuar a las entidades frente a una consulta.

Si se asume $P(e)$ uniforme, entonces el ranqueo de entidades solo depende de la estimación de $P(q|e)$. Se tiene lo siguiente:

$$score(e, q) = log(P(q|e)) = \sum_{t\in q} c(t, q)\ logP(t|\theta_e), \tag{3.1}$$

donde $c(t, q)$ denota la frecuenta del términos t en la consulta q y $P(t|\theta_e)$ es el modelo del lenguaje de la entidad. Este modelo se estima con la interpolación Jelinek-Mercer de la siguiente manera:

$$P(t|\theta_e) = (1 - \lambda)\ P(t|e) + \lambda\ P(t|\mathcal{E}), \tag{3.2}$$

[2]Este operador debería insertar términos nulos entre las subsecuencias de caracteres procesadas, de modo tal de separar los valores provenientes de distintas tripletas.

donde el parámetro λ controla la influencia del modelo de lenguaje de la colección. Las componente se calculan mediante *maximum likelihood*:

$$P(t|e) = \frac{c(t,e)}{l_e},\ P(t|\mathcal{E}) = \frac{\sum_{e\in\mathcal{E}} c(t,e)}{\sum_{e\in\mathcal{E}} l_e},$$

donde $c(t,e)$ es la frecuencia del término t en la descripción de la entidad y l_e es la longitud de la descripción de la entidad, es decir su número total de términos. También puede utilizarse el suavizado de Dirichlet para estimar $P(t|\theta_e)$.

A pesar de su simplicidad, los modelos de bolsas de palabras con representaciones no estructuradas de las entidades, generalmente tienen un sólido rendimiento, y son un buen punto de partida [9, p. 94]. Seguidamente, se adapta para la recuperación de entidades a los modelos del lenguajes con datos semiestructurados, donde las descripciones de entidades se componen de campos.

Modelos para descripciones de entidades semiestructuradas

Los Modelos de Lenguaje Mixtos (Sección 2.1.2) tienen en cuenta a los datos semiestructurados de los documentos. En este caso, se estima un modelo de lenguaje para cada campo f de e, y se los combina para formar un modelo lineal mixto de la entidad, es decir:

$$P(t|\theta_e) = \sum_{f\in\mathcal{F}} \alpha_f P(t|\theta_{f_e}), \tag{3.3}$$

siendo α_f el peso del campo f, de modo tal que $\sum_{f\in\mathcal{F}} \alpha_f = 1$, y θ_{f_e} el modelo de lenguaje del campo f de la entidad e.

Se puede estimar $P(t|\theta_{f_e})$ con el suavizado de Jelinek-Mercer, de la siguiente manera:

$$P(t|\theta_{f_e}) = (1-\lambda_f)\ P(t|\theta_{f_e}) + \lambda_f\ P(t|\theta_{f_\mathcal{E}}), \tag{3.4}$$

donde λ_f es el parámetro de suavizado del campo f. Las probabilidades son estimadas mediante *maximum likelihood*:

$$P(t|\theta_{f_e}) = \frac{c(t,f_e)}{l_{f_e}},\ \ P(t|\theta_{f_\mathcal{E}}) = \frac{\sum_{e\in\mathcal{E}} c(t,f_e)}{\sum_{e\in\mathcal{E}} l_{f_e}},$$

siendo $c(t,f_e)$ la frecuencia del término t en el campo f de la descripción de la entidad e y l_{f_e} la longitud del campo de la entidad.

Por otra parte, el Modelo Probabilista de Recuperación de Datos Semiestructurados (Sección 2.1.2) permite estimar α_f de manera no supervisada. Este peso se puede calcular basándose en la distribución de los términos del campo f de e. Formalmente, se reemplaza α_f de la Ecuación 2.10 con la probabilidad de *mapping*, $P(f|t)$,

$$P(t|\theta_e) = \sum_{f\in\mathcal{F}} P(f|t)P(t|\theta_{f_e}). \tag{3.5}$$

La estimación de $P(f|t)$ puede verse en la Sección 2.1.2.

En el estado arte suelen utilizarse representaciones de entidades semiestructuradas junto con rankings de entidades supervisados (*learning-to-rank*). Cabe destacar que los algoritmos learning-to-rank son supervisados, requieren un conjunto de entrenamiento.

3.3. Ranking de entidades enriquecido

Las bases de conocimiento organizan la información acerca de las entidades de una manera estructurada y semánticamente significativa. En esta sección, se muestra cómo preservar parte de las estructura de las entidades en una base de conocimiento, mediante un enfoque que tiene en cuenta representaciones de entidades semánticamente enriquecidas. Se describe un ranking de entidades enriquecido, basado no solo en los términos de la consulta, si no también en los tipos y entidades relacionadas a la consulta, tomando los métodos recopilados en [9].

Similitud basada en tipos

Las entidades en una base de conocimiento tienen tipos (clases). Si se tiene una consulta enriquecida, $\tilde{q} = (q, \mathcal{T}_q)$, siendo q la consulta en texto libre y $\mathcal{T}_q$ los tipos de la consulta, los cuales pueden estar dados por el usuario o identificados de manera automática a partir de la consulta [34], entonces las entidades pueden re-ranquearse de acuerdo a cuanto coincidan con los tipos de la consulta.

Así como se representó a las entidades como una distribución de probabilidad sobre términos, es posible crear una distribución sobre tipos. Sea $\theta_{\mathcal{T}_q}$ y $\theta_{\mathcal{T}_e}$ los modelos de tipos correspondientes a la consulta y entidad, respectivamente. La similitud basada en tipos puede calcularse mediante la divergencia KL [12]:

$$score_{type}(e, \mathcal{T}_q) = \max_{e' \in \mathcal{E}} KL(\theta_{\mathcal{T}_q} || \theta_{\mathcal{T}_{e'}}) - KL(\theta_{\mathcal{T}_q} || \theta_{\mathcal{T}_e}), \tag{3.6}$$

donde el máximo es utiliza para convertir divergencia KL en una similitud.

Los modelos de tipos se representan como distribución multinomial. Sea y un tipo, se tiene lo siguiente:

$$P(y|\mathcal{T}_e) = \frac{\mathbb{1}(y \in \mathcal{T}_e) + \mu_{\mathcal{T}} P(y|\mathcal{E})}{|\mathcal{T}_e| + \mu_{\mathcal{T}}},$$

donde $\mathbb{1}(y \in \mathcal{T}_e)$ toma el valor 1 si y es uno de los tipos asignados a e, si no es 0. El número total de tipos de e es $|\mathcal{T}_e|$, y al parámetro $\mu_{\mathcal{T}}$ se le asigna el promedio de tipos asignados todas las entidades del catálogo. El modelo de tipos del catálogo es la frecuencia relativa de los tipos sobre todas las entidades, es decir:

$$P(y|\mathcal{E}) = \frac{\sum_{e \in \mathcal{E}} \mathbb{1}(y \in \mathcal{T}_e)}{\sum_{e \in \mathcal{E}} |\mathcal{T}_e|}.$$

Mientras que, en el modelo de la consulta se distribuye la masa de probabilidad uniformemente sobre todos los tipos de la consulta:

$$P(y|\theta_{\mathcal{T}_q}) = \frac{\mathbb{1}(y \in \mathcal{T}_q)}{|\mathcal{T}_q|}.$$

En la siguiente sección, se muestra otro tipo de similitudes, donde se utiliza a las entidades asociadas a la consulta para mejorar su recuperación.

Similitud entre entidades

Si se tuviese un conjunto de entidades sobre las cuales se quisiera encontrar otras entidades similares, entonces se podría complementar la búsqueda textual con dicho conjunto, esto se llama *búsqueda basada en ejemplos* [9, p. 126]. Como antes, se requiere una consulta enriquecida, $\tilde{q} = (q, \mathcal{E}_q)$, donde $\mathcal{E}_q$ es el conjunto de entidades que sirven como ejemplos. Estos ejemplos pueden estar dados por el usuario o pueden obtenerse de

manera implícita, por ejemplo a partir un historial de búsqueda. Entonces, la estimación de la similitud basada en ejemplos puede hacerse utilizando un modelo como TF-IDF, el cual permite calcular la similitudes de documentos.

Sea una entidad e y una entidad ejemplo $e' \in \mathcal{E}_q$, la similitud basada en ejemplo se define de la siguiente manera:

$$score_{entity}(e, \mathcal{E}_q) = \frac{1}{|\mathcal{E}_q|} \sum_{e' \in \mathcal{E}_q} \text{sim}(e, e'). \tag{3.7}$$

La similitud entre pares de entidades puede realizarse de varias maneras. En este caso, se muestra la utilización del Modelo Espacio Vectorial. Sea $\mathbf{e}$ el vector de términos correspondientes a e:

$$\mathbf{e} = \langle w(t_1, e), \ldots, w(t_m, e) \rangle,$$

siendo m el tamaño del vocabulario, t_i términos del vocabulario y $w(t_i, e)$ un peso, por ejemplo tf-idf.

Luego, se comparan los vectores de términos con la similitud del coseno:

$$\text{sim}(e, e') = \frac{\mathbf{e} \cdot \mathbf{e}'}{\|\mathbf{e}\| \|\mathbf{e}'\|}.$$

Por último, las distintas representaciones de las entidades y la consulta enriquecida se combinan para formar una única función de ranqueo, de la siguiente manera [9, p. 104]:

$$score(e, \tilde{q}) = \lambda_{term}\ score_{term}(e, q) + \lambda_{type}\ score_{type}(e, \mathcal{T}_q) + \lambda_{entity}\ score_{entity}(e, \mathcal{E}_q),$$

siendo $\tilde{q} = (q, \mathcal{E}_q, \mathcal{T}_q)$, $score_{term}(e, q)$ la función de ranqueo basada en términos definida en la Sección 3.2 y las otras componentes son los modelos de recuperación vistos anteriormente.

En conclusión, es posible extender el ranqueo que utiliza a las representaciones de entidades como secuencias de términos para incluir información acerca de los tipos y entidades de interés para el usuario, las cuales pertenecen a una base de conocimiento. Para ello, no solo las representaciones de las entidades tienen que ser enriquecidas, si no también las consultas.

En la siguiente sección, nos preguntamos ¿cómo puede utilizarse la información de las bases de conocimiento (las entidades) para mejorar la recuperación de documentos?.

3.4. Expansión de consultas centradas en entidades

En general, los modelos de recuperación de documentos que supieron mejorar su efectividad, al incorporar a las bases de conocimiento, se pueden clasificar en tres grande grupos [9, p. 293]: (1) *basados en la expansión*, se utiliza a las entidades como fuentes para expandir la consulta con nuevos términos, (2) *basados en proyección*, donde la relevancia entre la consulta y los documentos se calcula proyectándolos a un espacio de entidades (un espacio "latente"[3]), (3) *basados en entidades*, se construye un espacio de entidades a partir de la información semántica explícita que tienen los documentos y las consultas, por ejemplo documentos anotados con entidades mencionadas, para aumentar la representación basada en términos.

[3]Se lo llama espacio de entidades latente por que la relación entre el texto y las entidades está latente (oculta) en la colección, de manera similar a lo que ocurre en los algoritmos de modelado de tópicos.

Es este trabajo se utiliza el primero de los enfoques, los modelos basados en la expansión mediante entidades, ya que si bien muestra menores rendimientos que los otros modelos, lo preferimos porque tenemos algoritmos no supervisados que encajan en esta categoría, como se el se describe a continuación. Entonces, seguidamente se describe un modelo que expande la consulta utilizando a las entidades en lugar de a los documentos.

Siguiendo la idea de Meij et al. [63], podemos formular a la expansión de la consulta como un doble proceso de traducción, primero se obtiene el conjunto de entidades relevantes y luego se utiliza el vocabulario de los términos asociados a las entidades de este conjunto como posibles términos para estimar el modelo expandido de la consulta. Podemos formalizarlo de la siguiente manera:

$$\begin{aligned} P(t|\hat{\theta}_q) &\propto \sum_{e \in \mathcal{E}} P(t|e,q)P(e|q) \\ &\approx \sum_{e \in \mathcal{E}} P(t|e)P(e|q), \end{aligned} \tag{3.8}$$

siendo $\mathcal{E}$ el conjunto de entidades. Este modelo recibe el nombre de Modelo de Lenguaje de Conceptos (*conceptual language models*).

La primera probabilidad $P(t|e)$, denominada **selección de términos**, nos indica cuales son los términos más importantes de la entidad e. La segunda probabilidad $P(e|q)$, la **selección de entidades**, identifica a la entidades a ser utilizadas en la expansión de la consulta, las cuales se corresponden con las entidades relevantes para q. En la práctica, solo los k términos con el puntaje más alto de la Ecuación 3.8 se tienen en cuenta para formar el modelo de la consulta expandida $\hat{\theta}_q$, con las probabilidades renormalizadas de modo tal de $\sum_t P(t|\hat{\theta}_q) = 1$.

En este modelo, se asume que $P(t|e,q) \approxeq P(t|e)$, es decir que se supone independencia condicional entre el término t y la consulta q. La razón por la que se prefiere $P(t|e)$ es porque su estimación puede realizarse de manera no supervisada, mientras que para estimar $P(t|e,q)$ se requieren de datos de entrenamiento. Sin embargo, asumir que todos los términos de la expansión son útiles y pueden beneficiar al rendimiento de recuperación, no sucede en la realidad, algunos términos son neutrales y otros pueden ser perjudiciales [9, p. 276].

Una forma sencilla de calcular $P(t|e)$ es por medio de los modelos de lenguaje de las entidades, es decir $P(t|e) = P(t|\theta_e)$. Mientras que, $P(e|q)$ se puede estimar de tres maneras [9, p. 271]:

- **Entidades mencionadas en la consulta**. Las entidades se identifican y desambiguan utilizando técnicas de *entity linking*.
- **Entidades recuperadas de una base de conocimiento**. Se utilizan las técnicas mostradas en la Sección 3.2 y se toman las primeras entidades. Sea $score_{ER}(e,q)$ el puntaje de relevancia para la entidad e sobre la consulta q (Ecuación 3.1) y $\mathcal{E}_q(k)$ las primeras k entidades relevantes a q. Luego, si normalizamos tenemos,

$$P(e|q) = \begin{cases} \frac{1}{Z} score_{ER}(e,q), & e \in \mathcal{E}_q(k) \\ 0, & sino \end{cases} \tag{3.9}$$

  donde Z es un coeficiente de normalización.
- **Entidades de pseudo-documentos relevantes**. No es necesario tener una representación explícita de la entidades para calcular $P(e|q)$, basta con tener a los

documentos anotados con entidades. Se consideran relevantes a las entidades que se mencionan en los primeros documentos que se obtienen por q [63].

De acuerdo a [99], utilizar *entity linking* para la sección de entidades es la opción más segura y la que mejor rendimiento alcanza. Sin embargo, tenemos pocas aplicaciones de reconocimiento de entidades en español y suelen ser de propósito general, no permiten identificar el vasto vocabulario legal que se requiere en esta tesina. Por lo tanto, se prefiere estimar la selección de entidades utilizando búsquedas de entidades sobre una base de conocimiento, como se muestra en la Sección 3.2, además esta opción permite utilizar todas las fuentes de información disponibles.

Recordemos que al modelo expandido de la consulta se lo combina con la consulta original:

$$P(t|\theta_q) = (1-\lambda)\frac{c(t,q)}{l_q} + \lambda P(t|\hat{\theta}_q). \tag{3.10}$$

Por último, se ranquean a la entidades utilizando la divergencia de KL con este nuevo modelo de la consulta, es decir:

$$score(e,q) = \sum_{t \in V} P(t|\theta_q)\ log\ P(t|\theta_e). \tag{3.11}$$

La eficacia de la expansión de la consulta centrada en entidades puede verse fuertemente afectada por el número de términos utilizados como expansión de la consulta [9, p. 292]. Se ha mostrado que este método supera de manera significativa al modelo de relevancia [100][63], siempre y cuando se tenga un buen método de selección de entidades. En general, para reducir los costos computacionales que implican la incorporación de información semántica de las bases de conocimiento, los modelos son implementados como un mecanismo de re-ranqueo. Primero se ranquean los documentos con métodos tradicionales como BM25, y luego los primeros k documentos son re-reanqueados utilizando modelos más avanzados.

En el próximo capitulo, se explica el tipo de fuente sobre la cual se emplean estos algoritmos de búsqueda, las bases de conocimiento o también llamadas ontologías.

Capítulo 4

Representación del conocimiento

El término *base de conocimiento* se utilizó por primera vez en el ámbito de los sistemas expertos. Estos sistemas, una de las primeras formas de software de IA, fueron diseñados para resolver problemas complejos, por ejemplo diagnósticos médicos, mediante el razonamiento sobre su conocimiento. El conocimiento para tales sistemas debe representarse de manera explícita, donde las bases de conocimiento deben contener una gran cantidad de aserciones sobre el mundo que describe a las entidades y sus relaciones.

En las últimas décadas, en los campos científicos y en la industria, ha crecido la adopción de las bases de conocimientos por el paradigma de la Web Semántica y los Datos Enlazados. La estandarización de lenguajes que definen y utilizan las bases de conocimiento —por ej. RDF(S), OWL, SPARQL—, generaron iniciativas que dieron acceso a ontologías y bases de conocimiento de numerosos dominio, como DBpedia, YAGO, etc.

Las **bases de conocimiento** contiene información acerca de entidades, clases semánticas, junto a sus atributos y relaciones. Esta puede dividirse en dos capas [6, p. 17]:

$$Base\ de\ conocimiento = TBox\ (esquema) + ABox\ (instancias)$$

- **TBox** (*terminological box*), donde se especifica el conocimiento intencional, es decir se describe el conocimiento general del dominio. Se definen las clases semánticas, las relaciones entre clases y entidades, propiedades que pueden tener las clases, y posiblemente restricciones y reglas.
- **ABox** (*assertional box*), se especifica el conocimiento al nivel extencional. Se compone de aserciones acerca de entidades específicas, describiendo sus nombres, tipos, atributos y relaciones con otras entidades.

Las bases de conocimiento capturan una representación útil del mundo (físico o virtual), con el objetivo de resolver un problema. El elemento básico de una base de conocimiento es la *entidad* [97]:

> Una **entidad** es un objeto abstracto o concreto de la ficción o la realidad.

Una entidad puede ser por ejemplo `exceso de velocidad`. Para describirla de manera no ambigua, es necesario que su nombre sea univoco, se requiere un identificador. En el modelo de datos de la Web Semántica, los identificadores toman la forma de URIs (identificador de recursos uniforme), una generalización de URLs. La URI de `exceso de velocidad` en el tesauro del SAIJ es la siguiente:

`http://vocabularios.saij.gob.ar/saij/xml.php?skosTema=12033`.

Una entidad también tiene *nombres* (*labels*) que son legibles para los humanos. El nombre de la entidad `exceso de velocidad` es "exceso de velocidad". Las entidades generalmente forman parte de grupos donde todos sus elementos tienen alguna característica en común. Se captura este conocimiento organizando a las entidades en *clases* o *tipos*.

> Una **clase** (o tipo) es el nombre de un conjunto de entidades que tienen una característica en común. Los elementos de la clase son llamados **instancias** de la clase.

Por ejemplo, `exceso de velocidad` es una entidad de la clase `infracción de tránsito` en la ontología legal desarrollada para esta tesina. Una entidad puede pertenecer a múltiples clases y las clases se pueden relacionar en términos de sus miembros mediante la disyunción, superposición, o la subsunción de una con otra. No siempre es obvio si algo debe modelarse como una entidad o una clase. Por ejemplo, podemos construir para cada entidad una clase con un único elemento que contenga a dicha entidad. Pero, en general, las clases contienen múltiples instancias [97].

Además, es posible relacionar dos clases especificando invariantes que deben cumplirse entre las clases, como la subsunción, también conocida como la relación **subclase**/**superclase**. Combinando estos invariantes, se construye una *jerarquía de clases*. Nos referiremos a este aspecto de la bases de conocimiento como una *taxonomía*.

> La clase A es una **subclase** de la clase B si todas las instancias de A son instancias de B.

Por ejemplo, la clase `señal de tránsito` es subclase de la clase `regla de tránsito`, por que toda señal de tránsito puede considerarse como una regla de tránsito.

No todas las bases de conocimiento, como veremos luego, hacen una clara distinción entre clases e instancias, ya que colapsan la relación `es-instancia-de` y `es-subclase-de` en una sola jerarquía `is-a`.

Por otra parte, las instancias pueden tener *propiedades*. Las bases de conocimiento las capturan en la forma de una relación matemática.

> Una **relación** para las instancias de las clases $C_1, \ldots, C_n$ es una subconjunto del producto cartesiano $C_1 \times \ldots \times C_n$, junto con un identificador para la relación.

Por ejemplo, si queremos decir que un documento legal, identificado como `suj0990829`, tiene como uno de sus temas centrales a los semáforos, podemos expresarlo como una relación:

$$\langle \texttt{suj0990829},\ \texttt{semaforo} \rangle \in \texttt{tieneTema},$$

donde `tieneTema` es el identificador de la relación.

La instancia de la relación `juriTieneConcepto` es una terna: el sumario, la catalogación y el concepto. El dominio de la relación `juriTieneConcepto` es `Documento_legal` × `ObjectProperty` × `Concept`, donde `Documento_legal` es la clase de los documentos legales (Sección 5.1), `ObjectProperty` es la clase de las propiedades en OWL (Sección 4.1.1) y `Concept` es la clase de los conceptos en SKOS (Sección 4.2.1).

Las instancias de una relación son comúnmente llamadas *hechos* o *declaraciones*. En la literatura, el término "relación" se lo usa para denotar tanto el identificador de la relación R como a la instancia $\langle x_1, \ldots, x_n \rangle$.

En otro ejemplo, podríamos describir el contenido textual del documento legal anterior de la siguiente manera:

$$\langle \texttt{suj0990829},\ \text{“}\textit{En nada obstaculiza la calidad de } \ldots\text{”}\rangle \in \texttt{tieneTexto}.$$

En este caso, se considera al texto del documento como un *valor* y no como una entidad, es decir que no tiene ninguna propiedad adicional. En el modelo de datos RDF, tales valores son llamados *literales*.

En general, bases de conocimiento utilizan exclusivamente a las relaciones binarias, y RDF tiene una notación formal para este caso, llamada **tripleta sujeto-predicado-objeto** (tripleta SPO).

> El modelo RDF restringe los roles en la tripleta SPO de la siguiente manera:
>
> - S debe ser una URI identificando a una entidad.
> - P debe ser una URI identificando una relación.
> - O debe ser una URI identificando a una entidad para una relación entre entidades, o un literal denotando el valor de un atributo.

Como las relaciones binarias pueden verse como un grafo etiquetado (un nodo para S y O y una arista con la etiqueta P), a las bases de conocimiento que ponen el foco en las tripletas SPO se las llaman **grafos de conocimiento**. Las tripletas suelen anotarse de la forma $\langle S, P, O\rangle$.

Otra característica que tienen las bases de conocimiento son restricciones lógicas y reglas. El propósito de las *restricciones* es asegurar la consistencia de la base de conocimiento, no pueden ingresarse declaraciones que violen las restricciones. Por ejemplo, podemos requerir que los temas que se asocien a los documentos legales mediante la relación `juriTieneConcepto` sean siempre conceptos de SKOS, esta sería una restricción de tipos. Mientras que, las *reglas* son útiles para derivar conocimiento adicional que lógicamente se puede deducir del contenido de la base de conocimiento.

Cuando la base de conocimiento esta más orientada al diseño de la TBox, se la suele llamar ontología, pero también se utiliza el término ontología para referirse a toda la base de conocimiento. [9, p. 38]. La ontología provee el esquema y las reglas para la interpretación de los individuos, hechos y otras reglas comprometidas en el conocimiento del dominio. Un grafo de conocimiento generalmente contiene la ontología y los datos relacionados [49, p. 8].

Seguidamente, en la Sección 4.1 se describen brevemente a las ontología, el lenguaje OWL para la representación del conocimiento y la herramienta semántica Entity Linking, la cual permite identificar a las entidades nombradas en un texto. Por otra parte, en la Sección 4.2 se tratan los tesauros, un Sistema de Organización del Conocimiento. Por último, en la Sección 4.3 se presentan algunas ontologías legales, principalmente el Tesauro SAIJ del Derecho Argentino.

4.1. Ontología

Se define una ontología de la siguiente manera [70]:

> Una ontología es una descripción formal y explicita de conceptos en un dominio del discurso (clases), propiedades de cada concepto describiendo varias características y atributos de los conceptos (propiedades) y restricciones (restricciones de roles).

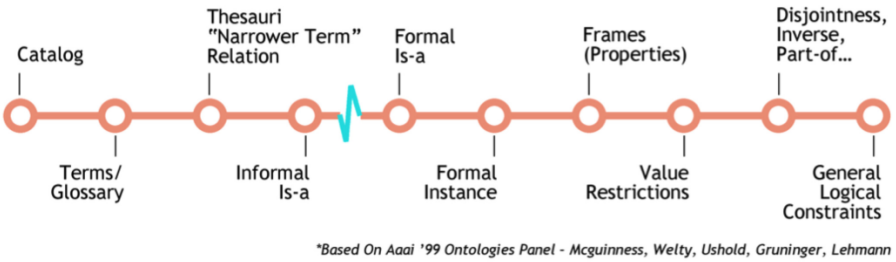


Figura 4.1: Espectro de expresividad y definición de una ontología (Fuente [49, p. 3]).

Las ontologías permiten la estandarización de la terminología de un dominio de aplicación, hace más fácil el intercambio de la información, tiene una sintaxis bien definida y una semántica formal basada en *lógicas descriptivas* (DL, *descriptive logic*). Presenta una visión intuitiva y unificada de las fuentes de datos, admite la formulación de consultas y provee de razonamiento automatizado. Las inferencias lógicas permiten aprovechar el conocimiento implícito para responder a las consultas [48].

En la Figura 4.1 se muestra el *espectro ontológico*, el rango de modelos de información que pueden ser llamados ontologías. Una ontología puede definirse en cualquier nivel del espectro, lo cual está determinado por los requisitos del negocio o aplicación [49, p. 3]. En la literatura, se distinguen a las ontologías que son principalmente taxonómicas de aquellas que modelan el dominio de un modo más profundo y proveen de mayores restricciones en su semántica. Se las llama *ontologías livianas* (*light ontology*) y *pesadas* (*hard ontology*), respectivamente, en la Figura 4.1 corresponden al lado izquierdo y al derecho. Las ontologías livianas incluyen taxonomías entre los conceptos, relaciones entre los conceptos y propiedades que describen a dichos conceptos. Por otra parte, las ontologías pesadas incluyen axiomas y restricciones, los cuales expresan el sentido intencional de los términos de la ontología [35, p. 8].

En la siguiente sección, se presenta el lenguaje OWL (*Web Ontology Language*) para la formalización de las ontologías en la Web.

4.1.1. OWL

El lenguaje OWL (*Web Ontology Language*) [2] es una recomendación de la W3C, permite definir ontologías en la Web Semántica. Provee clases, propiedades, individuos y datos que son almacenados como documentos de la Web Semántica. OWL facilita la interoperabilidad del contenido en la Web más allá de los formatos XML, RDF y RDF Schema (RDFs), al brindar más vocabulario junto con una semántica formal [62]. A grandes rasgos, OWL se puede expresar de la siguiente manera [103]:

$$OWL = RDFs + \text{nuevos constructores para una mayor expresividad.}$$

Comparado a RDF Schema, OWL provee la capacidad de expresar relaciones mucho más complejas y rica, con lo cual permite construir aplicaciones con una habilidad de razonamiento mayor. OWL primero fue estandarizado en 2004, y luego una nueva versión surgió en 2009, llamada OWL 2. OWL constituye una familia de lenguajes basados en lógicas descriptivas, los cuales son todos fragmentos del lenguaje más expresivo, OWL 2 DL. Las lógicas descriptivas (*description logics*, DL) son una familia de formalismos lógicos que representan un subconjunto de la lógica del primer orden. Fueron diseñadas para proveer un razonable grado de expresividad en el espectro ontológico, sin mucho poder de expresividad que dificulte razonamientos eficientes por parte de los razonadores.

Permite especificar a las ontologías en términos de conceptos (clases), roles (propiedades) e individuos (instancias) [49].

Como los lenguajes OWLs son un fragmento de la Lógica de Primer Orden, siguen la hipótesis *Open World Assumption* (OWA), es decir que la imposibilidad de derivar un hecho no implica lo contrario. Por ejemplo, si sabemos que Pedro es una persona, con esta información no podemos concluir que Pedro sea o no vegetariano [38]. El principio asume que la información es incompleta. Esto permite intencionalmente sub-especificar y permitir a otros extender y reutilizar la información. La hipótesis OWA sigue de la hipótesis *No Unique Name Assumption*, donde a menos que se especifique lo contrario no puede asumirse que recursos identificados por distintas URIs sean diferentes, por ejemplo Pedro y Juan quizás sean la misma persona.

Existen ciertas diferencias que hacen a OWL un lenguaje de la Web Semántica en comparación a los lenguajes regulares de DL, estas son[48]:

- OWL usa referencias URI como nombres, por ejemplo:

  `http://www.misitio.org/legalOnto#Antijuricidad`

  Se trata de la URI de la clase Antijuricidad en nuestra ontología.

- Tiene información sobre ontologías guardadas como documentos, incluye directivas como `owl:imports` para importar una ontología en otra.

- Añade tipos de datos de RDF y XML schema como rango de propiedades de datos.

- Diferente terminología: un concepto de DL es llamado **clase**, un rol de DL es llamado **object property**, y **data property** para los atributos.

En la Figura 4.2 pueden observarse aspectos sintácticos y semánticos de OWL 2. La sección superior muestra las distintas sintaxis que pueden utilizarse para serializar la ontología. En el centro, se representa la noción abstracta de la ontología, la cual puede pensarse como una estructura abstracta o un grafo RDF. En la sección inferior hay una semántica directa para los sub-lenguajes basados en OWL 2 DL y una semántica basada en RDF para OWL 2 Full.

En la siguiente sección se describe el procedimiento llamado *entity linking*, el cual enriquece el contenido textual mediante la anotación de entidades mencionadas con identificadores de un repositorio de conocimiento.

4.1.2. Entity linking

Entity linking es la tarea de reconocer las entidades mencionadas en el texto y asociarlas con las entradas correspondiente a un repositorio de conocimiento. Se lo llama repositorio de conocimiento por que suelen utilizarse fuentes de información menos estructuradas que las bases de conocimiento, por ejemplo Wikipedia [9, p. 152]. Los documentos semánticamente enriquecidos con entidades han probado ser útiles para una gran cantidad de tareas de procesamiento de texto, como la sumarización, poblado de bases de conocimiento, la recuperación de documentos, etc. En la tesina, se lo utiliza tanto para poblar a una ontología legal como para identificar a las entidades nombradas de una consulta.

A continuación, se define *entity linking* tomando como referencia el trabajo [9]. Dado un documento d, las entidades anotadas para el documento se describen como el conjunto $\mathcal{A}_d$, donde cada anotación $a \in \mathcal{A}$ es una tripleta, $a = (e, m_i, m_t)$, siendo e la

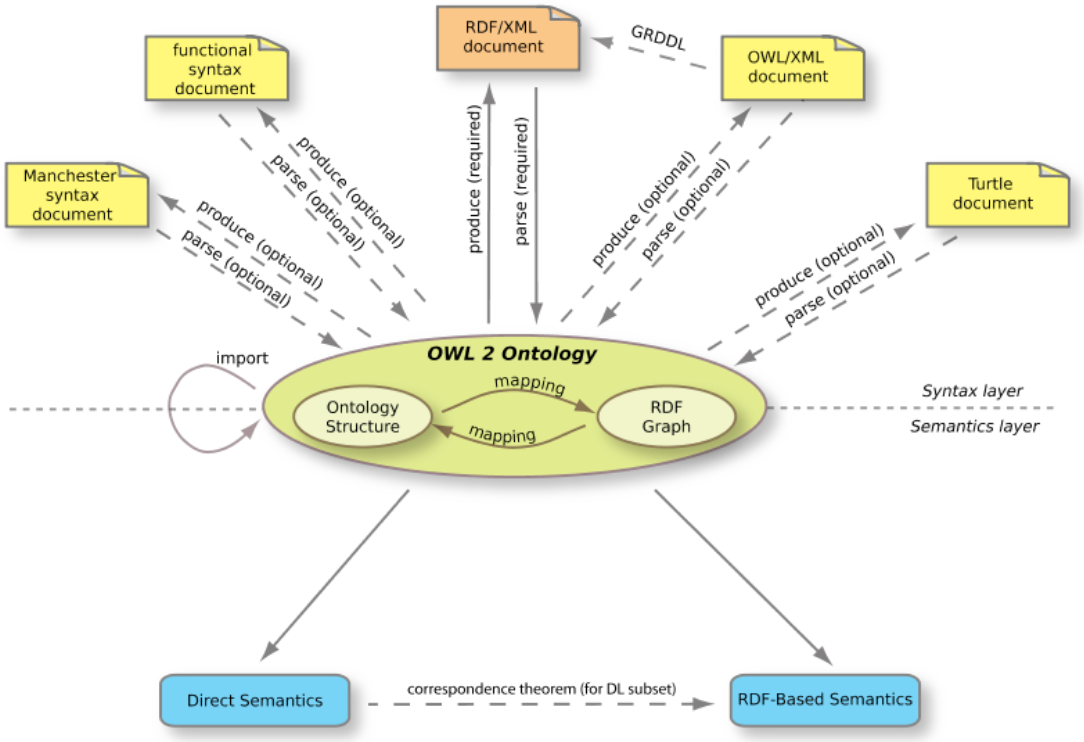


Figura 4.2: Sintaxis y semántica de OWL 2. (Fuente: `https://www.w3.org/TR/owl2-overview/`).

entidad —perteneciente al repositorio de conocimiento—, m_i y m_t los caracteres iniciales y terminales de la mención a e en d. Las menciones en $\mathcal{A}_d$ no deben superponerse.

Entity linking suele estar compuesto de las siguientes etapas:

- Detección de menciones: es responsable de identificar a las menciones (secuencias de términos contiguas) en un documento que se refieren a una entidad en particular. En general, se basa en un extenso diccionario con los nombres de entidades y sus variaciones.
- Selección de candidatos: una lista ranqueada de entidades candidatas es generada para cada mención.
- Desambiguación: se selecciona una entidad o ninguna para cada mención, basado en el contexto.

También, es posible efectuar solo dos etapas: la detección de entidades y la desambiguación. En este enfoque, la detección de menciones y la selección de candidatos es realizada en un solo paso.

La detección de entidades se realiza mediante un diccionario de entidades, $\mathcal{S}$, denominado *surface forms*, *name variants* o *aliases*. Se lo describe como una función $\mathcal{S} : s \to \mathcal{E}_s$, donde s es el nombre y $\mathcal{E}_s$ es el conjunto de entidades. Este diccionario debería ser extenso y contener variaciones comunes de los nombres de las entidades, como abreviaciones. Para la detección de las menciones, el documento es parseado y todas las cadenas de texto son chequeadas para detectar su presencia en $\mathcal{S}$, suelen utilizarse *tokens* de n-gramas de cierta longitud.

Luego, todas las entidades asociadas a una mención pueden ser candidatas. Si $\mathcal{E}_m$ es el conjunto de las entidades candidatas para una mención m, se puede decir:

$$\mathcal{E}_m = \{e \mid m \in \mathcal{S}_e\}.$$

Sin embargo, en la siguiente etapa, la selección de entidades candidatas, este conjunto suele acotarse para reducir los costos computacionales. Hasta aquí, se describió brevemente la detección de menciones. Para esta tesina se implementó un detector de menciones, que permite encontrar las entidades pertenecientes a la base de conocimiento legal (Sección 5.1) que son mencionadas en una consulta del usuario.

Los nombres de las entidades que conforman el *surface forms*, incluidas sus variantes, se obtienen de la base de conocimiento legal. Cuando las menciones se superponen, pueden tomarse distintos criterios, aquí se tomo el siguiente: las menciones subsumidas en otras menciones se ignoran. No se continuo con el desarrollo de un mecanismo de desambiguación, por dos motivos, una herramienta de *entity linking* no es el principal objetivo de esta tesina, y en la construcción de la detección de menciones legales usualmente encontramos a una sola entidad asociada a la mención, por lo tanto hay una sola entidad candidata para cada mención. Esto puede deberse a la poca cantidad de variantes de nombres en *surface forms*. De aquí en adelante, llamaremos a esta detección de menciones como *entity linking* de *match* exacto.

4.2. Tesauro

En la Bibliotecología y en las Ciencias de la Información se han desarrollo herramientas para organizar grandes colecciones de objetos, como libros u objetos de un museo. Se los conocen generalmente como Sistemas de Organización de Conocimiento (KOS, *Knowledge Organization Systems*), o también como vocabularios controlados. El término Sistema de Organización del Conocimiento se refiere a un amplio rango de esquemas —p. ej. encabezado de temas, tesauros, esquemas de clasificación— concebidos con diferentes propósitos y en diferentes momentos históricos. Sin embargo, todas ellos fueron diseñados para apoyar a la organización del conocimiento y la información de modo tal de facilitar su tratamiento y recuperación [61].

El objeto de esta sección son los tesauros utilizados para la recuperación de información, un tipo especial de KOS. La función principal de un tesauro es ayudar en la recuperación de información guiando la elección de los términos para la indexación y búsqueda [24]. Una de sus principales aplicaciones es la *indexación temática* (*subject indexing*), en donde el indexador asigna un conjunto de descriptores a los documentos, y el usuario puede utilizarlos para formar su consulta.

En el estándar de tesauros ISO 25964 [44], se define al tesauro como "un vocabulario controlado y estructurado, donde los conceptos son representados por términos, organizado de manera tal que las relaciones entre los conceptos sean explícitas, y cuyos términos preferidos estén acompañados por sinónimos o cuasisinónimos". Los términos pueden ser preferidos, cada concepto puede tener uno solo por lenguaje, o no preferidos, cada concepto puede tener más de uno por lenguaje. La relación de cada concepto con su nombre se anota mediante USE (para término preferido) o UF (para término no preferido).

Una característica intrínseca de los tesauros es su habilidad para distinguir y mostrar la relación estructural entre los conceptos. Se distinguen dos tipos de relaciones en los tesauros: (1) relaciones a nivel micro, utilizadas para representar las equivalencias, jerarquías y relaciones asociativas entre conceptos, (2) relaciones a nivel macro, encargadas de

Relación	Significado
BT	*Broader term*, se relaciona con un concepto más arriba en la jerarquía, debe tener un significado más amplio o menos específico.
NT	*Narrower term*, propiedad inversa de BT.
RT	*Related term*, refiriéndose a un concepto relacionado, sin ser concepto BT ni NT.

Tabla 4.1: Relaciones de un tesauro.

HORSES
uf *equus caballus*
BT1 equidae
BT2 perissodactyla
BT3 mammals
BT4 vertebrates
BT1 livestock
BT2 domestic animals
BT3 animals
NT1 draught horses
NT1 foals
NT1 mares
NT1 racehorses
NT1 saddle horses
NT2 ponies
NT1 stallions
NT2 geldings
rt meat animals

Figura 4.3: Entrada para un término preferido de la versión en inglés de AGROVOC edición 1982 [55].

relacionar grupos de sujetos/facetas. Estas facetas pueden llamarse "temas", "dominios", "subconjuntos" o "microtesauros", los conceptos se organizan de acuerdo a las categorías que representan, por ej. tiempo, lugar, procesos, etc. [59].

Las relaciones jerárquicas *broader term* (BT) y *narrower term* (NT) proveen la estructura del tesauro que se establece entre los conceptos, las cuales deberían entenderse como "broader concept" y "narrower concept"[1]. Para las relaciones asociativas entre dos conceptos que no están en la misma jerarquía pero son semánticamente similares, se utiliza *related term* (RT). En el Cuadro 4.1 se describe el uso de estas relaciones.

En la Figura 4.3 puede verse una entrada del tesauro AGROVOC que tiene como término preferido a *horses*. Este es un caso de polijerarquía, ocurre cuando un concepto tiene más de un concepto general (*broader*).

A continuación, se describe al formato SKOS que permite representar un Sistema de Organización del Conocimiento en la Web Semántica.

[1]Esto generó confusión acerca de los roles entre los términos y los conceptos, ya que el estándar ISO 2788, en 1974, pretendía relaciones entre conceptos, pero sugiriendo términos en su notación BT/NT/RT. Como esta convención era ampliamente utilizada, se decidió no cambiarla, por lo tanto esta notación se mantiene en los estándares hasta la actualidad [29].

4.2.1. SKOS

SKOS [65] (*Simple Knowledge Organization System*) es una recomendación de la W3C, se trata de un vocabulario y un modelo de datos para expresar Sistemas de Organización del Conocimiento en la Web Semántica.

Los sistemas KOS están definidos informalmente, en general no fueron diseñados con una representación formalmente precisa del dominio del conocimiento. En consecuencia, un sistema KOS no pueden traducirse a lenguajes de la Web Semántica como RDFS y OWL, con implicaciones formales lógicas, sin un gran esfuerzo de modelado. Por esta razón se diseño SKOS, con una semántica básica e informal que permite expresar distintos tipos de sistemas KOS. Se considera a SKOS como un paso intermedio, un puente entre el caos resultante del bajo nivel de estructuración de la Web actual y el riguroso formalismo descriptivo de las ontologías definidas con OWL [92]. SKOS proporciona un vocabulario muy sencillo y un modelo intuitivo que puede ser utilizado conjuntamente con OWL o de manera independiente.

SKOS entiende a KOS como un conjunto de conceptos, que pueden agruparse en un *concept scheme*. Cada concepto y *concept scheme* es un recurso, que puede identificarse por una URI. Un concepto representa una idea y puede tener varios *nombres* o *labels* asociados, los cuales son literales que pueden ser preferidos o no preferidos.

Cada *concept scheme* es un conjunto de uno o más conceptos SKOS. Relaciones semánticas entre estos conceptos también pueden verse como parte del *concept scheme*. En general, cada KOS es considerado un *concept scheme*. Conceptos del mismo KOS pueden relacionarse por relaciones jerárquicas, *brother than*, *narrower than*, o por relaciones asociativas, *related*. Las colecciones de conceptos son grupos ordenados de conceptos que pueden llevar *labels*. Éstas son útiles para agrupar conceptos que tienen algo en común, y que es conveniente agruparlas bajo un nombre en común, o donde los conceptos pueden tener algún orden [92].

SKOS es una ontología OWL Full (indecidible). Pero, también se propusieron pequeños cambios que modifican a SKOS para ser una ontología OWL DL (decidible)[2].

Las relaciones en SKOS tienen una semántica ambigua, su significado está expresado informalmente en su notación, `skos:broader` y `skos:narrower` indican que pueden interpretarse como las relaciones "es más amplio" y "es más especifico" [4, p. 309]. Es decir que, no sabemos si `skos:narrower` se trata de una relación *es-instancia-de*, *es-subclase-de* o *es-parte-de*. Por ejemplo, un `saxo soprano` *es-parte-de* una `banda de jazz`, esto no significa que cada saxo soprano *es-instancia-de* banda de jazz. Por ello, se dice que SKOS colapsa *es-instancia-de*, *es-parte-de*, y *es-subclase-de* en una sola relación general. El grupo de trabajo que definió el estándar de SKOS discutió si deberían definir subpropiedades de `skos:broader`, como `broaderGeneric` (para la subsunción de clases), `broaderInstantive` (para instanciación de clases) y `broaderPartitive` (relación *parte de*), como se hizo en el estándar de tesauro ISO 25964. Concluyeron que esto resultaría en una superposición entre RDFs y OWL. Se podría ver a `broaderGeneric` como un equivalente a `rdfs:subClassOf` y `broaderInstantive` a `rdf:type` [8].

Esta es la principal diferencia entre un tesauro y una ontología, los KOSs son utilizados para organizar la información, mientras que las ontologías son utilizadas para representar el conocimiento [104]. El conocimiento "explícito" en una ontología formal se expresa a través de axiomas y hechos. Pero un tesauro o cualquier tipo de esquema de clasificación no incluye este tipo de afirmaciones, sino que identifica y describe (con el lenguaje natural o expresiones no formales) ideas o significados a los que nos referimos

[2]Una discusión acerca de los axiomas que causaron que SKOS fuese interpretado como OWL Full se encuentra en `http://swig.hpclab.ceid.upatras.gr/SKOS/Skos2Owl2`

como conceptos. Estos conceptos pueden organizarse en estructuras que carecen de una semántica formal y que no pueden considerarse como axiomas o hechos. Por ejemplo, "el concepto X tiene como nombre preferido a 'Y' y es parte del tesauro 'Z"', son hechos sobre el tesauro, pero no son hechos sobre la forma en la que el mundo esta organizado en un dominio en particular, como podría expresarse en una ontología formal. Es decir, un tesauro únicamente proporciona un mapa intuitivo de como están organizados los temas dentro de procesos de clasificación y búsqueda de objetos (generalmente documentos) relevantes a un dominio específico [65]. Sin embargo, podemos referirnos a los sistema KOS como ontologías livianas, en consecuencia SKOS puede considerarse como un lenguaje para definir ontologías livianas.

Para convertir un tesauro o esquema de clasificación en conocimiento formal, debe transformarse a una ontología, un proceso que puede resultar muy costoso. Sin embargo, los tesauros pueden ser útiles como estructuras informales y convenientes para la navegación dentro de un dominio temático. Utilizarlos de esta manera no requiere ninguna reingeniería y, por lo tanto, es mucho menos costoso [65]. Por lo tanto, el trabajo adicional y la complejidad necesaria para convertirlo en una ontología deben equilibrarse con los beneficios previstos para la aplicación [44].

4.3. Ontologías legales

Las ontologías han sido utilizadas en el ámbito legal para una gran variedad de aplicaciones, nos interesa resaltar aquellas que han sido desarrolladas para la estandarización del contenido de textos regulatorios y sus metadatos. La adopción de estándares web como XML, RDF, SPARQL, y la publicación de la información como datos enlazados dieron paso al diseño de vocabularios y ontologías sobre documentos legales. Haremos un breve repaso de los esfuerzos para publicar la legislación y jurisprudencia como Datos Enlazados Abiertos por parte de varios estados.

Entre las ontologías impulsadas por un organismo internacional, se destaca *European Legislation Identifier* (ELI), diseñada con el fin de unificar las legislaciones nacionales de la Unión Europea. La ontología ELI [1] se basa en el modelo de datos establecido en Requisitos Funcionales de los Registros Bibliográficos[3], donde se distinguen el concepto "obra" (creación intelectual o artística), "expresión" (la realización intelectual o artística de una obra) y "manifestación" (la encarnación física de una expresión). En el caso de ELI, se definen las siguientes clases:

- Recurso legal (LegalResource): un recurso normativo —p. ej. norma o artículo—.
- Expresión legal (LegalExpression): la realización concreta de un recurso legal, generalmente una secuencia de caracteres alfanuméricos.
- Formato (Format): puede tratarse de un formato electrónico o papel.

Otro aspecto que tiene en cuenta esta ontología es la indexación temática de los documentos legales. En la ontología ELI, se define la propiedad `eli:is_about` con dominio en LegalResource y rango en SKOS, para asignarle a un recurso legislativo un concepto SKOS, utilizando preferentemente al tesauro EuroVoc.

También, tenemos un esquema de metadatos para la jurisprudencia de los estados europeos llamado *European Case Law Identifier* (ECLI), el cual define un conjunto de propiedades Dublin Core para anotar a los casos legales. En contraste con ELI, no es una

[3] `https://www.ifla.org/`.

ontología formal, sino más bien una recomendación de nueve propiedades obligatorias y ocho propiedades opcionales que pueden utilizarse para describir los metadatos de los documentos.

Existen varios portales europeos que implementan a ELI para expresar los metadatos de sus normas con este estándar. Además, encontramos iniciativas que extienden y reutilizan a la ontología ELI:

- Lynx[4]: es un proyecto de la Unión Europea para definir una base de conocimiento legal y proveer servicios como la sumarización de documentos, búsqueda de legislación, anotación de documentos mediante *entity linking*, etc. La base de conocimiento legal se define en base a ELI y NLP Interchange Format[5] (NIF) [30].
- Semantic Finlex[6]: se propone dos ontologías para la publicación de legislación y jurisprudencia finlandesa como Datos Abiertos Enlazados [72]. Por un lado, la ontología *Semantic Finlex Legislation* extiende a la ontología ELI, y por el otro, la ontología *Semantic Finlex Case Law* reutiliza el modelo de metadatos ECLI para definir un ontología similar a ELI.
- Nomothesia[7]: se desarrolla un buscador experimental para la legislación griega, donde los documentos se modelan con una ontología que extiende a MetaLex y ELI. Además, el contenido de los documentos es enlazado con la ontología DBpedia [22].

En Latinoamérica, la Biblioteca del Congreso Nacional de Chile desarrolló una ontología sobre los recursos legislativos del congreso chileno[8] —p. ej. normas, proyectos de ley— [90]. En la siguiente sección, se presenta al tesauro del SAIJ y se detallan los tipos de búsquedas que brinda el portal del SAIJ.

4.3.1. SAIJ: Sistema Argentino de Información Jurídica

El Sistema Argentino de Información Jurídica (SAIJ) tiene más de 970 000 documentos judiciales almacenados en su base de datos [14]. Contiene documentos del ámbito nacional, provincial e internacional, de una gran variedad: leyes, jurisprudencia, doctrina, etc, actualizados diariamente. Por ejemplo, el repositorio incluye todas las leyes nacionales, sancionadas desde 1853. Además, tanto las leyes, jurisprudencia como doctrina, en su mayoría, se encuentran catalogadas manualmente con temas del Tesauro SAIJ del Derecho Argentino, como se dijo anteriormente, este procedimiento es llamado indexación temática.

El SAIJ provee búsquedas temáticas y búsquedas por palabras claves, estas se analizan a continuación.

El Tesauro SAIJ del Derecho Argentino y la búsqueda por facetas

El Tesauro SAIJ del Derecho Argentino (TSDA) fue desarrollado para indexación temática de los documentos del Sistema Argentino de Información Jurídica (SAIJ). Este tesauro fue elaborado teniendo en cuenta la información contenida en la base de datos del SAIJ y las necesidades de información del usuario, por ello se lo describe como una

[4] `https://lynx-project.eu/`.
[5] La ontología NIF tiene como propósito la anotación de texto con información lingüística.
[6] `http://data.finlex.fi`.
[7] `http://legislation.di.uoa.gr/`.
[8] `https://datos.bcn.cl/ontologies/bcn-resources/doc/index.es.html`.

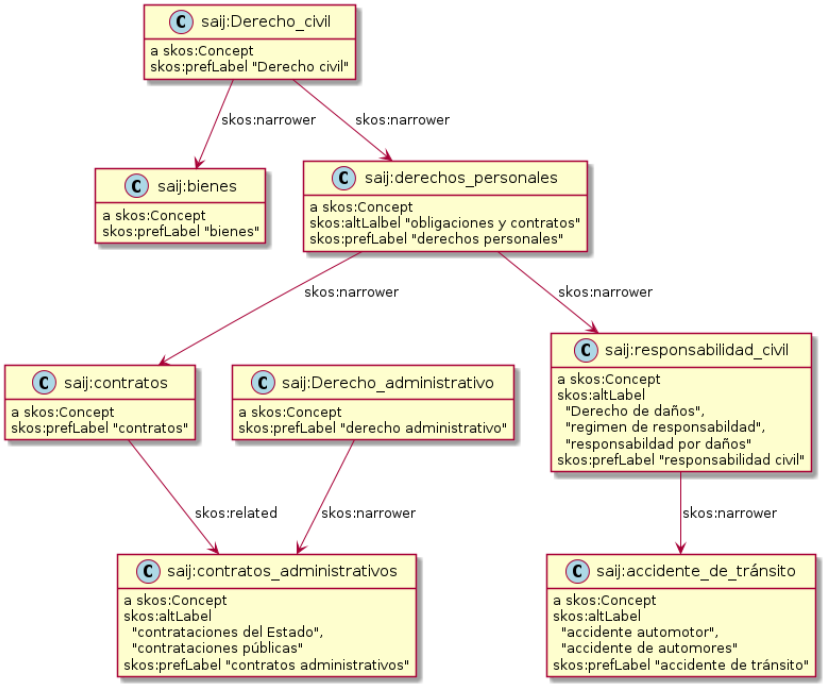


Figura 4.4: Fragmento del Tesauro SAIJ del Derecho Argentino.

herramienta pragmática: "La clasificación de las voces no se ajustó necesariamente a ordenamientos doctrinarios rígidos, sino que se tuvieron en cuenta las necesidades del usuario de la base de datos y su forma de interpretar la información, donde se podrán encontrar voces tan específicas como vidrio roto o netamente jurídicas como daño en bienes de uso público" [14].

El tesauro se encuentra disponible para su descarga[9] en formato SKOS, es desarrollado mediante el software de gestión de vocabularios controlados tematres[10] [36]. Se encuentra organizado en facetas que responden a un área temática específica (Derecho Civil, Derecho Penal, Bienestar Social, Salud Pública, etc.) [14]. Se compone de más de 21.000 conceptos, donde se categoriza al derecho en 17 grandes ramas: Derecho Administrativo, Derecho Aeronáutico, Derecho Ambiental, Derecho Civil, Derecho Informático, etc., y otros 10 temas que no son exclusivamente del derecho como Salud Pública, Transporte, Seguridad Social, Recursos Naturales, etc., en total hay 27 conceptos *top*. Cuando la taxonomía del tesauro es diseñada siguiendo el principio del análisis por facetas se lo llama *tesauro facetado* [95]. En la Figura 4.4 se muestra una pequeña parte del tesauro, donde se puede ver que cada concepto tiene asociado su términos preferido, y en ciertas ocasiones términos alternativos.

Las *facetas* se corresponden a una dimensión (característica) que es relevantes en la colección de documentos a la cual se le quiere incorporar la búsqueda por facetas (o navegación facetada), por ejemplo en la Figura 4.5 se pueden ver las facetas del SAIJ, *Tipo de documento*, *Fecha*, *Organismo*, *Tribunal*, etc. Cada faceta tiene una jerarquía de términos (categorías o valores) asociada, por ejemplo los términos de la faceta *Tipo de documento*, son *Jurisprudencia*, *Legislación*, *Dictamen*, *Doctrina* y *Novedad*, además se muestra para cada categoría un contador, que indica cuantos documentos fueron clasificados en con la categoría. La asignación de cada documento de la colección con las categorías de las

[9] `http://datos.jus.gob.ar/dataset/tesauro-saij-de-derecho-argentino`
[10] `http://admin.tcda.infojus.gov.ar/saij/index.php`

Figura 4.5: Facetas iniciales del SAIJ.

Figura 4.6: Valores de las facetas luego de filtrar los resultados de la Figura 4.5 de acuerdo al valor *Jurisprudencia*.

facetas recibe el nombre de *clasificación facetada*. Esto permite a los usuarios explorar la colección mediante las facetas, como filtros. Por ejemplo, si el usuario filtra la búsqueda por el término *Jurisprudencia* de la faceta *Tipo de documento*, que aparece en la Figura 4.5, entonces todos los resultados corresponden a la jurisprudencia, ver Figura 4.6 donde se muestra cuales serían las facetas retornadas sobre esta segunda búsqueda.

En el caso de la faceta *Tema* la mayoría de su jerarquía de términos proviene del tesauro del SAIJ. Es decir, se utiliza al Tesauro del SAIJ como la faceta *Tema*. La clasificación facetada y la navegación facetada son ampliamente utilizada en la web, una de las claves de su éxito ha sido su integración con la búsqueda de palabras claves.

La búsqueda por facetas fue desarrollada por Hearst et al.[40] en el proyecto Flamenco. Tiene algunas deficiencias como volverse confusa si no reflejan el modelo mental que tiene el usuario, los documentos no están asignados a los términos de las facetas apropiadamente, o las facetas son muy "extensas" y/o "profundas" [41]

A partir del tesauro del SAIJ, se propone la construcción de una base de conocimiento legal, como una fuente información del sistema de búsqueda desarrollado en esta tesina, para más detalles ver la Sección 5.1. Otro tipo de búsqueda en el SAIJ es la búsqueda por texto libre o palabras claves, se la describe en la siguiente sección.

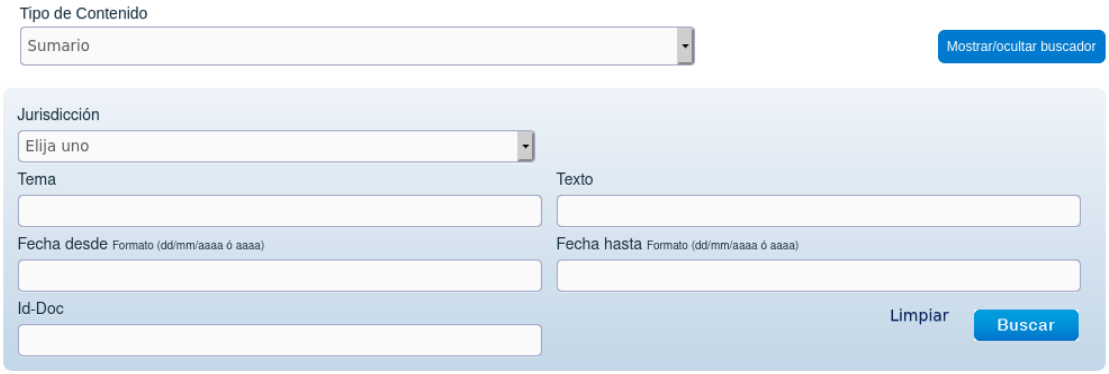


Figura 4.7: Interfaz de búsqueda del SAIJ.

Búsquedas por palabras claves

El SAIJ implementa la búsqueda en texto libre desde la caja de búsqueda *Texto*, ver Figura 4.7. Como se dijo anteriormente, motores de búsqueda como Lucene, permite ingresar una búsqueda en texto libre, la cual es parseada como una consulta booleana con el fin de filtrar los resultados, para luego ranquearlos con un modelo *best match*. Desconocemos cuál es el modelo de recuperación implementado en SAIJ, pero desde la interfaz podemos analizar cómo se parsean las búsquedas, es decir, conocer su expresión booleana.

Cada término ingresado en la caja de búsqueda es tratado como una conjunción, es decir que la consulta del usuario se traduce a una consulta booleana de operadores ANDs. Por ejemplo, la consulta embestimiento motocicleta se parsea de la siguite manera:

```
embestimiento AND motocicleta.
```

La búsqueda de texto libre implementada en el SAIJ es estrictamente conjuntivas, desde la interfaz sólo pueden ejecutarse conjunciones, no es posible expresar explícitamente ningún otro operador booleano en la consulta. Sin embargo, puede expresarse una consulta booleana directamente desde la url, utilizando los operadores AND, OR, NOT, también admite búsqueda por frases. Es decir, la interfaz del SAIJ no implementa todas las variantes de búsqueda que soporta su motor de búsqueda. Por ejemplo, podemos expresar la siguiente consulta desde la url:

```
embestimiento OR motocicleta.
```

Las limitaciones en las búsqueda del SAIJ, búsquedas estrictamente conjuntivas y facetas que pueden alcanzar una gran profundidad, motivaron el desarrollo de un sistema de búsqueda semántico que incorpore las fuentes de información de este caso. Con la misma información de un sistema de búsqueda facetado, es posible sugerir conceptos relacionados a la búsqueda de texto libre del usuario y mejorar la cantidad de resultados relevantes, incorporando expansiones de consultas. En el siguiente capítulo se detalla el sistema propuesto.

Capítulo 5

Sistema propuesto

Para lograr un mejor entendimiento de la necesidad de la información del usuario pueden aplicarse distintos métodos de expansión, auto-completado o sugerencias a las consultas. La mayoría de los métodos de auto-completado y sugerencia de consultas —p. ej. *query log mining*— requieren de un historial de consultas o estadísticas sobre el comportamiento del usuario mientras este interactúa con el sistema de búsqueda. En este caso, no se cuenta con dicha información, pero se tiene a disposición una base de conocimiento y documentos indexados semánticamente. Por lo tanto, pueden aplicarse métodos de expansión de expansión, como la retroalimentación por (pseudo) relevancia.

Incorporar en un sistema de búsqueda la información contenida en una base de conocimiento es un problema abierto y la búsqueda centrada en entidades ha mostrado alcanzar buenos rendimientos en dicho sentido. De acuerdo a la fuentes de información disponibles, un tesauro y una colección de documentos indexados temáticamente, se escogió y extendió el Modelo de Lenguaje de Conceptos. La utilización de un modelo basados en entidades más complejos posiblemente requiera de mayor información, es decir, documentos anotados con las entidades mencionados o una gran cantidad de juicios de relevancia.

Se proponen dos modelos de búsquedas semánticos y no supervisados[1] que utilizan la información contenida en un base de conocimiento para expandir la consulta, la cual debe ser del mismo dominio de la colección de documentos. La expansión se realiza mediante la retroalimentación por (pseudo) relevancia basada en entidades. Por un lado, el Modelo de Relevancia con Entidades, genera de manera automática las entidades que se esperan sean relevantes a la consulta, mediante el uso de las descripciones de entidades. Luego, expande la consulta con los términos más importes de las entidades generadas. El otro método, el Modelo Iterativo de Relevancia con Entidades, requiere la asistencia del usuario para que seleccione entre las entidades sugeridas, en una o más iteraciones, aquellas que considere relevantes.

Un sistema de búsqueda que implemente alguno de estos modelos debería mantener una arquitectura como la que se muestra en la Figura 5.1, a continuación se describe cada una de sus componentes.

Cuando el usuario ingresa una búsqueda de texto libre se la transforma en términos índice y a partir de estos términos se calcula el algoritmo de búsqueda. El modelo de recuperación retorna una lista ranqueada de documentos que finalmente es mostrada al usuario como respuesta a su consulta [27]. Esta transformación de la entrada y salida del sistema *online* de búsqueda se lleva a cabo en la capa **interacción con el usuario**.

[1] La función de ranking no se entrena a partir de un conjunto de juicios de relevancia como los algoritmos *learning-to-rank*.

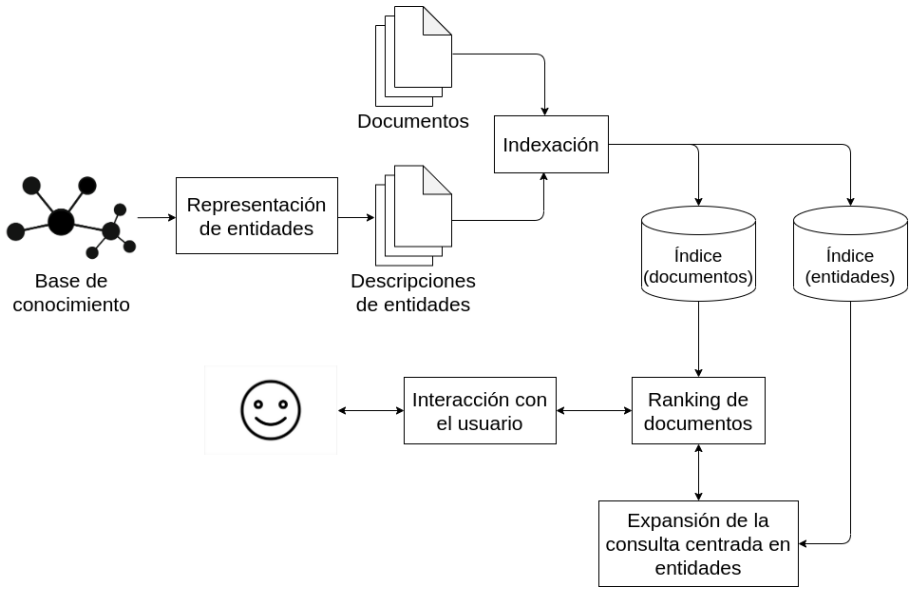


Figura 5.1: Arquitectura propuesta.

Cuando se desea expandir la consulta, es necesaria su reformulación por parte del modelo de expansión de la consulta y sobre esta consulta reformulada se ejecuta el algoritmo de búsqueda. A diferencia de una expansión tradicional, el algoritmo de expansión utiliza los términos de las entidades pertenecientes a una base de conocimiento, en lugar de los documentos de la colección. El núcleo del motor de búsqueda, la implementación del modelo de recuperación y expansión de la consulta, se encuentran en la capa **ranking de documentos** y **expansión de la consulta centrada en entidades**, respectivamente.

Como se dijo, la expansión puede realizarse de manera automática o iterativa. Si se utiliza el Modelo Iterativo de Entidades de Relevancia, se genera a una lista ranqueada de entidades, como sugerencias de búsqueda, en cada iteración del modelo. Estas se muestran al usuario y éste es quien debe juzgar si son relevantes a su consulta inicial. En la última iteración se procede a la reformulación de la consulta. Los detalles de su funcionamiento se verán más adelante.

El sistema propuesto debe contar con dos fuentes de información, una colección de documentos y una base de conocimiento. Sin embargo, el modelo de expansión de la consulta no opera directamente con la base de conocimiento, si no que actúa sobre los términos de las descripciones de las entidades. Estos documentos se crean a partir de la base de conocimiento, mediante el procedimiento *predicate folding*. La capa **representación de entidades** se encarga de implementar dicha transformación.

Entonces, el sistema de búsqueda cuenta con dos colecciones de documentos:

- La colección de documentos que se quiere recuperar.
- La colección de documentos o descripciones que representan a las entidades de la base de conocimiento.

Los sistema de búsqueda necesitan precomputar ciertos valores estadísticos de su colección de documentos para realizar los cálculos de manera eficiente, los cuales se mantienen en estructuras de datos llamadas índices (Sección 1.1.1). En este caso, es necesaria la construcción de dos índices, uno por cada colección de documentos. La capa **indexación** se encarga de realizar la normalización e indexación de los documentos de manera *offline*. El vocabulario de términos del sistema de búsqueda es la intersección

de los vocabularios de cada índice. El modelo de expansión de la consulta se encarga de consultar el índice de entidades, mientras que el modelo de recuperación de documentos consulta al índice de los documentos.

Se aplica este sistema de búsqueda al dominio legal, en particular al derecho civil, para mejorar la precisión de los resultados que puede encontrar un abogado es su búsqueda de jurisprudencia relevante para la construcción del marco legal de un caso. También, se busca proveer de sugerencias de búsqueda que permitan guiar al usuario en la formación de su consulta. Se utiliza como fuente de información a una base de conocimiento legal, diseñada especialmente para esta tesina, y una colección de sumarios, recolectados para este trabajo. Comenzaremos, en la siguiente sección, describiendo la construcción de la base de conocimiento dedicada al ámbito legal.

5.1. Desarrollo de una base de conocimiento legal

Se distinguen dos tipos de datos: los documentos y las entidades, se tratan de una colección de documentos y una base de conocimiento, respectivamente. En la recuperación de entidades generalmente se utilizan bases conocimientos de propósito general —p. ej, DBpedia—, donde cada entidad se asocia a un gran número de propiedades, esto beneficia su recuperación, mediante los métodos vistos en la Sección 3.2, ya que cuando mayor es el conocimiento acerca de una entidad, mayor es la información disponible para la creación de su descripción. La descripción de una entidad X es el documento que contiene la información que se conoce de X en la base de conocimiento, como se mostró en la Sección 3.1. Si solo tomáramos al tesauro del SAIJ como fuente de información del sistema de búsqueda, la expansión de la consulta añadiría sinónimos y conceptos relacionados/generales/específicos de los conceptos mencionados en la consulta, ya que esta es la única información disponible en el tesauro. Pero, además del tesauro, contamos con documentos (sumarios) de la colección anotados con conceptos del tesauro del SAIJ, es decir que los documentos están indexados semánticamente. Una manera de incluir esta información en el sistema de búsqueda, es definiendo una base de conocimiento que modele esta situación. Por lo tanto, se diseñó una ontología para la indexación semántica de documentos legales con conceptos del estándar SKOS. Entonces, se tienen dos fuentes de información: el tesauro del SAIJ y la ontología mencionada. También, se exploró la posibilidad de enriquecer al tesauro del SAIJ con más conceptos y nuevas funcionalidades, a partir del uso constructores previstos en el estándar de tesauros ISO 25964 [44], pero que no fueron implementados en el estándar SKOS, nos referimos a los grupos de conceptos transitivos y a especializaciones de propiedades propias del dominio legal. Se extendió al tesauro del SAIJ sobre un tema en particular, los accidentes de tránsito.

Se combinó la información proveniente del tesauro del SAIJ, el tesauro de los accidentes de tránsito y la ontología sobre la indexación semántica de documentos legales en una sola base de conocimiento, la llamamos *LegalBase*. Esta nueva base de conocimiento describe tanto la indexación semántica de los documentos legales como su organización conceptual. Por último, *LegalBase* se pobló automáticamente con los sumarios de la colección.

De esta manera, se creo una base de conocimiento legal para ser utilizada como fuente de expansión de la consulta. Esta base de conocimiento permite identificar cuáles son los conceptos legales que trata cada sumario y cómo se organizan, según el tesauro del SAIJ y el tesauro de los accidentes de tránsito. Con esta información se crean las descripciones de las entidades legales, como se describe en la Sección 5.2.

A continuación, se detalla la construcción del tesauro de los accidentes de tránsito.

Tesauro de accidentes de tránsito

Se crea de forma manual un pequeño tesauro sobre los accidentes de tránsito. Su utilización en el sistema propuesto podría traducirse en mejores respuestas para las consultas sobre este tipo de casos. Se inicia la creación del tesauro identificando un conjunto de palabras claves que podría utilizar un abogado para describir su necesidad de información si este afrontase un caso sobre accidentes de tránsito. En esta etapa se cuenta con la asistencia de un abogado. Las palabras claves se obtienen de casos ficticios y del tesauro del SAIJ. Luego, se define un concepto SKOS para cada una de las palabras claves que se general en el paso anterior y se les da una estructura con las propiedades que provee el estándar SKOS, estas son `skos:broader`, `skos:narrower` y `skos:related`. Se define un total de 91 conceptos SKOS, donde 11 son conceptos nuevos que no pertenecen al tesauro del SAIJ. Es decir, se reutilizaron conceptos del SAIJ en la construcción de este nuevo tesauro, los conceptos del SAIJ mantienen sus URIs originales, no se le asignan URIs nuevas. De esta manera, se evitar tener que desarrollar una alineación de conceptos entre los tesauros.

El tesauro de los accidentes de tránsito tiene 5 conceptos *top*. Los conceptos *top* son conceptos que no tienen conceptos más generales (*broader*) [2]. Estos son: "accidente de tránsito", "regla de tránsito", "transporte", "infracción de tránsito", "derecho civil". Por ejemplo, el concepto *top* "regla de tránsito" tiene 5 conceptos más específicos: "casco reglamentario", "cinturón", "luces reglamentarias", "velocidad reglamentaria" y "señal de tránsito".

Hasta aquí, se tiene lo que llamamos un tesauro tradicional de los accidentes de tránsito. Seguidamente, se explica la necesidad de extenderlo con la incorporación de grupos de conceptos.

Ante un caso de accidente de tránsito, lo que busca la víctima es una indemnización y por esta razón su abogado debería probar la existencia de los siguientes tópicos:

- Daño.
- Antijuricidad. Es decir que, se produjo una acción contra el ordenamiento jurídico (las leyes).
- Relación de causalidad.
- Reproche o factor de atribución sobre el demandado.

A partir de este ejemplo, notamos que ciertos conceptos del tesauro podrían "encasillarse" en temas como: *daño*, *relación de causalidad* y *factor de atribución* —p. ej., el concepto "lucro cesante" podría agruparse como *daño*—. Es decir, ciertos temas podrían tratarse como conjuntos de conceptos. Esta funcionalidad es llamada **grupo de conceptos** en el estándar de tesauros ISO 25964.

En la literatura [3], se describen dos tipos de relaciones en un tesauro: a nivel micro, la equivalencia, jerarquía y relaciones asociativas entre los conceptos, y a nivel macro, las relaciones entre temas o grupos de facetas, también llamados "dominios", "subconjuntos" o "microtesauros"[3]. La agrupación de los conceptos por temas suele realizarse si el tesauro tiene una amplia gama de dominios. Los tesauros puede tener múltiples niveles de grupos de conceptos. Por ejemplo, el tesauro EuroVoc tiene 21 temáticas y 127 microtesauros,

[2] En SKOS, es posible declarar a un concepto como *top* de un tesauro utilizando la propiedad `skos:topConceptOf`.

[3] Un subconjunto del tesauro capaz de funcionar como un tesauro [45].

el dominio "Ambiente" es dividido es estos tres microtesauros: "política medioambiental", "medioambiente" y "deterioro del medioambiente" [59].

En ISO 25964, los grupos de conceptos pueden tener su propia relación jerárquica, mediante las propiedades *hasSubGroup*/*hasSuperGroup*[4]. Los conceptos pertenecientes a un grupo de conceptos pueden tener una relación de jerarquía o asociativa con otros conceptos [98]. En SKOS tenemos constructores similares a los grupos de conceptos [59], estos son: `skos:Collection` y `skos:ConceptSchema`, los cuales se definen como conjuntos de conceptos. Se prefiere utilizar a `skos:Collection`, ya que las colecciones de SKOS pueden anidarse mediante la propiedad `skos:member`, de manera similar a *hasSuperGroup*. En cambio, los objetos `skos:ConceptScheme` no pueden relacionarse. La manera de agregar conceptos a los colecciones es también con la propiedad `skos:member`. De aquí en adelante, trataremos a las colecciones y grupos de conceptos como sinónimos.

En el siguiente ejemplo se utiliza la sintaxis Turtle [79]. Se declaran las colecciones *tránsito*, *señal de tránsito* y *regla de tránsito*, y se les añaden conceptos.

```
@prefix ex1: <http://example1.com/> .
@prefix ex2: <http://example2.com/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .

ex2:regla_de_tránsito rdf:type skos:Collection ;
  skos:member ex1:casco_reglamentario ;
  skos:member ex1:cinturón_de_seguridad ;
  skos:member ex1:regla_de_tránsito .

ex2:señal_de_tránsito rdf:type skos:Collection ;
  skos:member ex1:semáforo ;
  skos:member ex1:señal_de_tránsito .

ex1:señal_de_tránsito rdf:type skos:Concept .
ex1:semáforo  rdf:type skos:Concept .
ex1:regla_de_tránsito rdf:type skos:Concept .
ex1:casco_reglamentario rdf:type skos:Concept .
ex1:cinturón_de_seguridad rdf:type skos:Concept .
```

En el ejemplo, el concepto "señal de tránsito" y la colección "señal de tránsito", son dos entidades distintas. Para evitar que tengan la misma URI, tanto en el tesauro como en este ejemplo, son declaradas en distintos espacios de nombres. En este caso, los conceptos se encuentran en el espacio de nombres `ex1` y las colecciones están en `ex2`. En el tesauro, se utilizan los espacios de nombres `accidenteDeTránsito`, `legalCollection` y `legalConcept`. Pero, aquí evitamos utilizarlos y en su lugar llamamos a los dos primeros como `ex1` y `ex2`, simplemente por cuestiones de espacio. Continuando con el ejemplo, se les añaden las relaciones `skos:narrower` a los conceptos ya declarados de la siguiente manera:

```
ex1:regla_de_tránsito skos:narrower ex1:casco_reglamentario  .
ex1:regla_de_tránsito skos:narrower ex1:cinturón_de_seguridad .
ex1:señal_de_tránsito skos:narrower ex1:semáforo .
```

[4]Puede verse el modelo de datos ISO 25964 en `http://www.niso.org/schemas/iso25964/Model_2011-06-02.jpg`

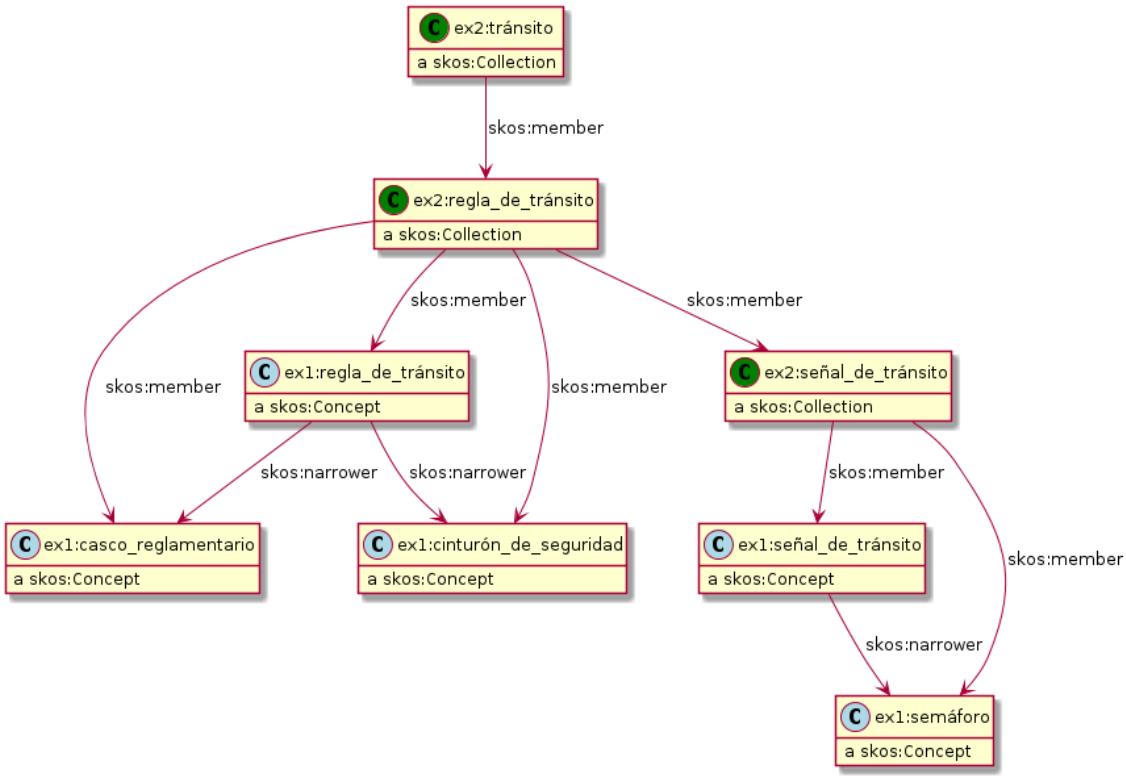


Figura 5.2: Diagrama del ejemplo que muestra el uso de los grupos de conceptos.

Se puede decir que el grupo "señal de tránsito" es un subgrupo de "regla de tránsito" y éste es a su vez un subgrupo de "tránsito" de la siguiente manera:

```
ex2:tránsito skos:member ex2:regla_de_tránsito .
ex2:regla_de_tránsito skos:member ex2:señal_de_tránsito  .
```

En la Figura 5.2, puede verse el grafo RDF que se obtiene de este ejemplo. Por otra parte, en la Figura 5.3, se muestra toda la taxonomía de grupos que se definió para el tesauro de los accidentes de tránsito. El grupo *raíz* del cual descienden todos los demás, es el grupo *Legal*. Este grupo se refiere a todos los términos creados por la ley —p. ej., contrato— y aquellos términos del mundo alcanzados por la ley, —p. ej., accidente de tránsito—, un enfoque similar a [52].

Los grupos de conceptos permite encontrar conceptos similares, con alguna característica en común, a partir de la identificación de conceptos de un mismo grupo. Si existe una jerarquía de grupos, es posible inferir la similitud de conceptos entre grupos ancestros. Esta característica se detalla en la siguiente sección.

Grupos de conceptos transitivos

En ISO 25964, los grupos pueden tener una jerarquía mediante las propiedades *hasSubGroup*/*hasSuperGroup* y pueden considerarse transitivas, es decir "todos los miembros de un grupo X son miembros de los grupos ancestros de X"[5]. Siguiendo el ejemplo anterior, ¿podemos inferir que el concepto *semáforo* es miembro del grupo de conceptos o

[5]En la ontología ISO-THES [91], que extiende SKOS para incluir los constructores del estándar ISO 25964, se define a la propiedad *subGroup* y se la anota con el siguiente enunciado: "todos los miembros de subGroup (objeto) son miembros del grupo (sujeto)", pero no se la declara como transitiva.

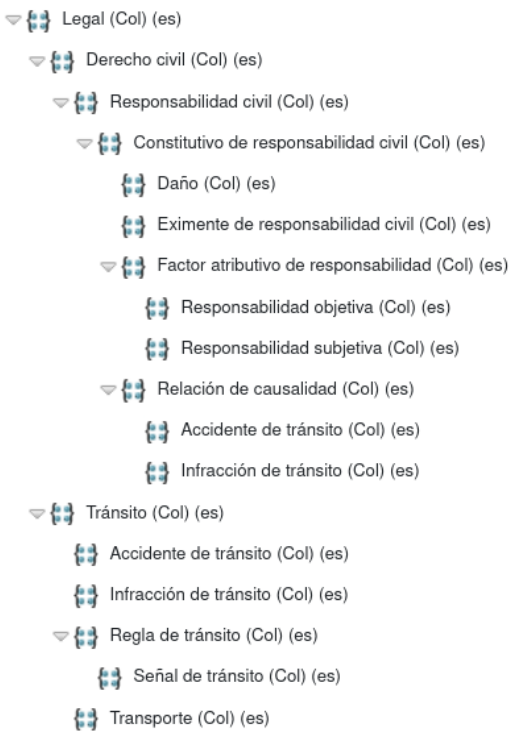


Figura 5.3: Taxonomía de las colecciones legales.

la colección *regla de tránsito*? no es posible, ya que `skos:member` no es una propiedad transitiva en el estándar SKOS. Si bien las colecciones de SKOS pueden anidarse, no son transitivas.

La transitividad nos permitiría decir que los conceptos no son sólo "similares" a los conceptos declarados en su mismo grupo si no también a los conceptos declarados en grupos ancestros. Una manera de añadirla sería cambiando la definición de `skos:member` de modo de hacerla transitiva. En principio, cuando creamos una nueva ontología reutilizando otras ya disponibles, podemos extender, especializar o adaptar la ontología que es reutilizada [76], de modo tal de satisfacer nuestros requerimientos. Pero, se desalienta la inclusión de axiomas que cambien la inferencia de instancias fuera de nuestro espacio de nombres, es decir, si se tienen datos de SKOS, al realizar la inferencia se debería obtener lo que propone el estándar. Cuando una nueva ontología redefine la semántica (clases, propiedades) de otra ontología, se lo llama *ontology hijacking* [43]. Para evitar este problema, decidimos expresar la transitividad de una manera implícita, utilizando un patrón de diseño llamado *Class-Individual Mirror* [4, p. 345], el cual permite representar a un individuo como una clase. La razón principal para utilizar este patrón es la posibilidad de incluir clases, las cuales pueden utilizarse por algoritmos de búsqueda de entidades que trabajan sobre este tipo de dato, como el que se describe en la Sección 3.3. Se muestra a continuación la manera de incorporar la transitividad con este patrón siguiendo el ejemplo de la sección anterior.

Primero, se define la inversa de `skos:member`, con dominio en `skos:Concept`, se la anota como `skos:conceptMemberOf`,

```
ex1:memberOf a owl:ObjectProperty ;
        owl:inverseOf skos:member .
```

```
ex1:conceptMemberOf a owl:ObjectProperty ;
   rdfs:subPropertyOf ex1:memberOf ;
   rdfs:domain skos:Concept ;
   rdfs:range skos:Collection .
```

Luego, por ejemplo, se representa a la colección `n2:señal_de_tránsito` como a una clase, se define a `Concepto_en_colección_señal_de_tránsito` como la clase de todos los individuos que son miembros de la colección `n2:señal_de_tránsito`, mediante la restricción `owl:hasValue`, de la siguiente manera:

```
ex1:Concepto_en_colección_señal_de_tránsito rdf:type owl:Class ;
  owl:equivalentClass [ rdf:type owl:Restriction ;
  owl:onProperty ex1:conceptMemberOf ;
  owl:hasValue ex2:señal_de_tránsito
  ] .
```

Recordemos que anteriormente se declaró la siguiente tripleta,

```
ex2:señal_de_tránsito rdf:type skos:Collection ;
  skos:member ex1:semáforo ;
  skos:member ex1:señal_de_tránsito .
```

De esta manera, se puede inferir lo siguiente, se utiliza * para distinguir a las tripletas inferidas:

```
* ex1:semáforo rdf:type ex1:Concepto_en_colección_señal_de_tránsito .
* ex1:señal_de_tránsito rdf:type ex1:Concepto_en_colección_señal_de_tránsito .
```

También es posible hacer el razonamiento inverso, si supiésemos lo siguiente:

```
ex1:semáforo rdf:type ex1:Concepto_en_colección_señal_de_tránsito .
ex1:señal_de_tránsito rdf:type ex1:Concepto_en_colección_señal_de_tránsito .
```

Se puede inferir:

```
* ex2:señal_de_tránsito skos:member ex1:semáforo ;
    skos:member ex1:señal_de_tránsito .
```

Si aplicamos el patrón, definiendo la clase Y que describe a todos los conceptos que son miembros del grupo X, mediante una restricción en la propiedad `ex1:conceptMemberOf`, entonces todo concepto miembro del grupo X también pertenecerá a la clase Y, y viceversa. A continuación, se definen otras clases siguiendo el patrón.

```
ex1:Concepto_en_colección_regla_de_tránsito a owl:Class ;
  owl:equivalentClass [ rdf:type owl:Restriction ;
  owl:onProperty ex1:conceptMemberOf ;
  owl:hasValue ex2:regla_de_tránsito
  ] .

ex1:Concepto_en_colección_tránsito a owl:Class ;
  owl:equivalentClass [ rdf:type owl:Restriction ;
  owl:onProperty ex1:conceptMemberOf ;
  owl:hasValue ex2:tránsito
  ] .
```

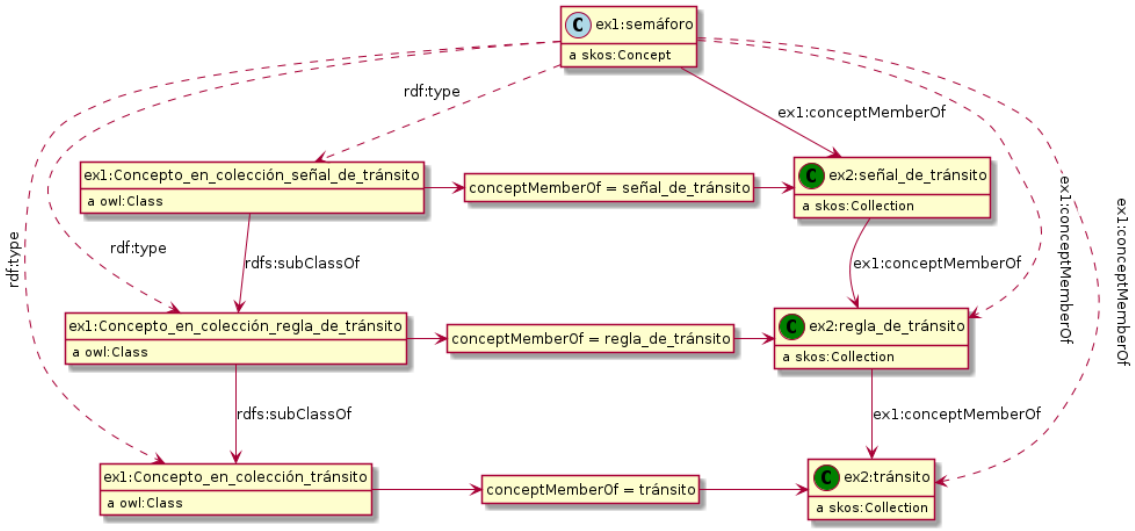


Figura 5.4: Aplicación del patrón *Class-Individual Mirror*.

Luego, la transitividad de las colecciones se consigue si se define la taxonomía de clases equivalente a la taxonomía de las colecciones. En este caso, debemos expresar lo siguiente: todo concepto que sea un *concepto en la colección señal de tránsito* será un *concepto en la colección regla de tránsito* y un *concepto en la colección regla de tránsito* será un *concepto en la colección transito*, de la siguiente manera:

```
ex1:Concepto_en_colección_señal_de_tránsito
  rdfs:subClassOf ex1:Concepto_en_colección_regla_de_tránsito .
ex1:Concepto_en_colección_regla_de_tránsito
  rdfs:subClassOf ex1:Concepto_en_colección_tránsito .
```

Ahora, es posible inferir la transitividad de `skos:member`:

```
* ex2:regla_de_tránsito skos:member ex1:semáforo ,
    ex1:señal_de_tránsito .
* ex2:tránsito skos:member ex1:caso_reglamentario ,
    ex1:cinturón_de_seguridad ,
    ex1:regla_de_tránsito ,
    ex1:semáforo ,
    ex1:señal_de_tránsito .
```

En la Figura 5.4, se muestran con líneas punteadas las inferencias que pueden realizarse sobre el concepto "semáforo", luego de haber aplicado el patrón.

El patrón *Class-Individual Mirror* modela situaciones donde no se tiene en claro si algo debería modelarse como una clase o individuo: ¿debemos refinos a él como un conjunto de individuos o como una cosa por si misma? [4, p. 365]. Este patrón modela ambas situaciones y las mantiene sincronizadas. Adiciones en el lado del individuo se reflejan en el lado de las clases y viceversa. Se la utilizó para expresar la transitividad de `skos:member` sin declararla explícitamente como transitiva, además permite ver a los grupos de conceptos como clases. Los modelos de búsqueda propuestos hacen uso de las clases de este tesauro al incluir, en la selección de entidades, la similitud de tipos declarada en la Sección 3.3.

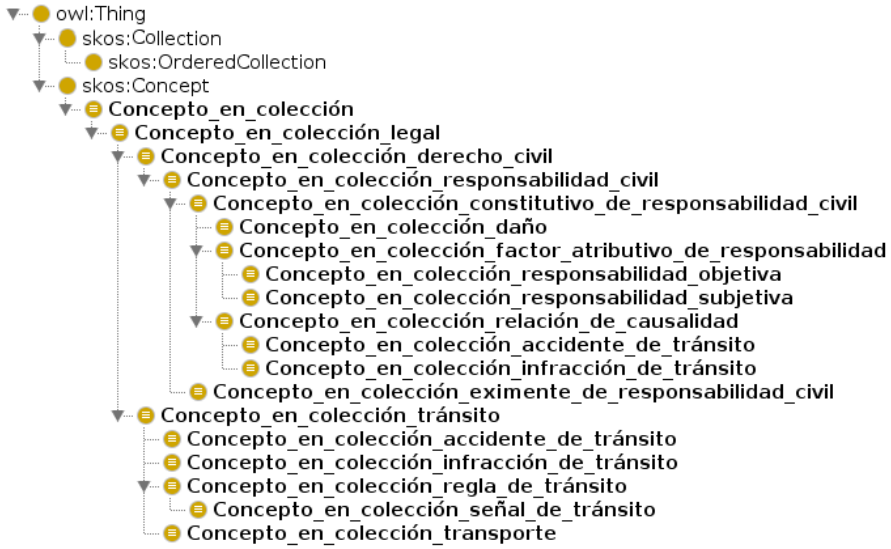


Figura 5.5: Taxonomía de las clases de grupos de conceptos del tesauro de accidentes de tránsito.

En la Figura 5.5, se puede ver la a taxonomía de clases sobre los grupos de conceptos, isomorfa a la taxonomía de grupos vista en la Figura 5.3. En la siguiente sección, se explica una última extensión al tesauro de los accidentes de tránsito, nuevas propiedades.

Especializaciones de las relaciones de SKOS

Cuando se reutilizan elementos de una ontología pueden producirse algunos problemas. Por ejemplo, si en el tesauro de los accidentes de tránsito se declara lo siguiente:

```
ex1:deber_de_obrar_con_prudencia skos:broader ex1:imprudencia .
```

Mientras que en tesauro del SAIJ, se tiene:

```
ex1:imprudencia skos:broader ex1:deber_de_obrar_con_prudencia .
```

Cuando se integren ambos tesauros, se obtendrá un ciclo en la jerarquía *skos:broader* y si bien es consistente con respecto a las reglas de integridad de SKOS, puede representar un problema para las aplicaciones que utilicen estos sistemas de organización del conocimiento [65]. Entonces, se decide distinguir claramente las relaciones jerárquicas y asociativas que tiene cada tesauro. En lugar de utilizar las relaciones `skos:broader`, `skos:narrower` y `skos:related` se definen nuevas sub-propiedades en el tesauro de los accidentes de tránsito, `ex1:tieneGeneral`, `ex1:tieneEspecifico` y `ex1:seRelaciona`, como se muestra a continuación.

```
ex1:seRelaciona rdf:type owl:ObjectProperty ;
  rdfs:subPropertyOf skos:related ;
  rdf:type owl:SymmetricProperty .

ex1:tieneEspecifico rdf:type owl:ObjectProperty ;
  rdfs:subPropertyOf skos:narrower ;
  owl:inverseOf :tieneGeneral .
```

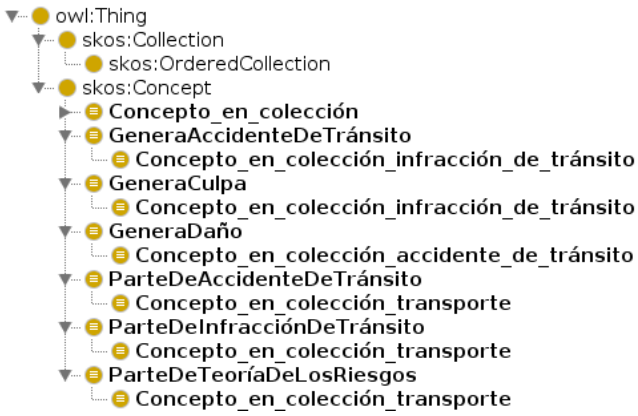


Figura 5.6: Taxonomía de las restricciones en las propiedades `ex1:puedeGenerar` y `ex1:esParteDe`.

```
ex1:tieneGeneral rdf:type owl:ObjectProperty ;
  rdfs:subPropertyOf skos:broader .
```

Además, se declaran las especializaciones `ex1:esParteDe` y `ex1:puedeGenerar`, de la siguiente manera:

```
ex1:esParteDe rdf:type owl:ObjectProperty ;
  rdfs:subPropertyOf ex1:tieneGeneral .

ex1:puedeGenerar rdf:type owl:ObjectProperty ;
  rdfs:subPropertyOf ex1:seRelaciona ;
  rdf:type owl:TransitiveProperty .
```

La propiedad `ex:puedeGenerar` indica la posibilidad de que una situación dada por el concepto objeto pueda "ocurrir" si se produce la situación descripta por el concepto sujeto. Así, se pretende construir una relación de causalidad (transitiva).

Se decidió utilizar las especializaciones `ex1:esParteDe` y `ex1:puedeGenerar` sobre todos los conceptos de algunas colecciones, p. ej. *todo concepto en la colección "infracción de tránsito" puede generar un accidente de tránsito*. Para formalizar el enunciado anterior, se define `ex1:GeneraAccidenteDeTránsito` como la clase de todos los individuos que pueden generar un accidente de tránsito, utilizando la restricción `owl:hasValue`:

```
ex1:GeneraAccidenteDeTránsito rdf:type owl:Class ;
  owl:equivalentClass [ rdf:type owl:Restriction ;
  owl:onProperty ex1:puedeGenerar ;
  owl:hasValue ex1:accidente_de_tránsito
  ] .
```

Luego, la clase `ex1:Concepto_en_colección_infracción_de_tránsito` pasa a ser una subclase de `ex1:GeneraAccidenteDeTránsito`,

```
ex1:Concepto_en_colección_infracción_de_tránsito
  rdfs:subClassOf ex1:GeneraAccidenteDeTránsito .
```

Entonces, las clases que representan a algunos grupos de conceptos, también son descriptas por las propiedades `ex1:esParteDe` y `ex1:puedeGenerar`. En la Figura 5.6 se

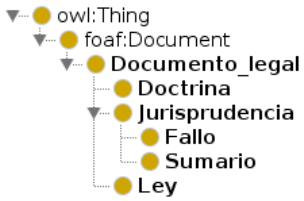


Figura 5.7: Taxonomía de los documentos legales.

tiene una taxonomía de clases, donde se representa que los conceptos de ciertos grupos generan o son parte de otro concepto.

De esta manera, quedan definidas las principales componentes del tesauro de los accidentes de tránsito, enriquecido con grupos de conceptos transitivos y con ciertas especializaciones en las relaciones semánticas de SKOS. En la siguiente sección, se presenta a una ontología que relaciona a los documentos legales con conceptos del estándar SKOS.

Una ontología de documentos legales

En la Sección 4.3 se describieron ontologías legales utilizadas principalmente por entidades gubernamentales para formalizar tanto el contenido textual de sus documentos como sus metadatos. Luego, este conocimiento podría ser explotado con herramientas semánticas por ejemplo, con el fin de mejorar la recuperación de documentos. Pero, la mayoría se diseñaron teniendo en cuenta a un solo tipo de documento legal, las normas legales, mientras que la jurisprudencia generalmente no fue considerada. En lugar de extender a alguna de estas ontologías, se decidió desarrollar una pequeña ontología sobre documentos legales, que incluya a la legislación, jurisprudencia y doctrina. En esta ontología, se definen a los documentos legales de la siguiente manera:

```
@prefix legalOnto: <http://www.semanticweb.org/legalOnto> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

legalOnto:Documento_legal rdf:type owl:Class ;
  rdfs:subClassOf foaf:Document .
```

La clase `foaf:Document` proviene del estándar FOAF[6], de acuerdo a su documentación, "representa todas las cosas que son generalmente concebidas como documentos". Como se hizo en el tesauro de los accidentes de tránsito, reutilizamos vocabulario ya existente, lo cual es muy importante para permitir la interoperabilidad de las aplicaciones computacionales[7]. En la Figura 5.7 puede verse el resto la taxonomía de los documentos legales, se declararon las clases *doctrina*, *jurisprudencia* y *ley*.

Otro aspecto importante a modelar es la asociación entre los documentos y los temas que tratan, es decir la indexación semántica. En la tesina, conocer cuales son los sumarios clasificados con X concepto legal permite la incorporación dichos sumarios en la descripción de la entidad X, como se muestra en la Sección 5.2.

[6]FOAF es una ontología para describir personas, sus actividades y sus relaciones con otras personas y objetos. El nombre *Fried of a Friend* evoca la relación que se presenta en las redes sociales.

[7]Encontramos esta misma reutilización en la ontología del Congreso Nacional de Chile, donde se define a la clase Leyes como subclase de `foaf:Document`.

En el estándar SKOS no se incluye un mecanismo para la indexación temática, pues sus desarrolladores consideraron que ya existían relaciones de indexación, por ejemplo en Dublin Core[8] se tiene la propiedad `dct:subject`[9]. En este caso, se prefiere definir una nueva propiedad de indexación, llamada `legalOnto:tieneTema`, en lugar de utilizar `dct:subject`.

A continuación, puede verse la definición de `legalOnto:tieneTema` y su propiedad inversa `legalOnto:esTemaEn`.

```
legalOnto:tieneTema rdf:type owl:ObjectProperty ;
  rdfs:domain legalOnto:Documento_legal ;
  rdfs:range skos:Concept .

legalOnto:esTemaEn rdf:type owl:ObjectProperty ;
  owl:inverseOf legalOnto:tieneTema ;
  rdfs:domain skos:Concept ;
  rdfs:range legalOnto:Documento_legal .
```

Por último, se definen las propiedades `legalOnto:texto` y `legalOnto:titulo`, para describir el contenido de un documento —el texto— y su título —en general palabras claves dadas por los catalogadores del SAIJ—, se hace de la siguiente manera:

```
legalOnto:texto rdf:type owl:DatatypeProperty ,
                owl:FunctionalProperty ;
  rdfs:domain legalOnto:Documento_legal ;
  rdfs:range xsd:string .

legalOnto:titulo rdf:type owl:DatatypeProperty ,
  owl:FunctionalProperty ;
  rdfs:domain legalOnto:Documento_legal ;
  rdfs:range xsd:string .
```

Esta ontología modela la indexación semántica de documentos legales mediante conceptos SKOS, la llamamos *legalOnto*.

Integración

A partir de las bases de conocimiento descriptas previamente, es decir, el tesauro de los accidentes de tránsito, la ontología de documentos legales y el tesauro del SAIJ, se crea una única fuente de información de modo tal de centralizar todo el conocimiento disponible sobre las entidades legales. El mecanismo utilizado para la integración es implementado por la directiva *owl:import*. Su semántica es la siguiente, si una ontología *OntoA* importa a *OntoB*, entonces todas las tripletas pertenecientes a *OntoB*, así como las tripletas pertenecientes a las ontologías importadas por esta ontología, pertenecen a *OntoA*. Este tipo de reutilización recibe el nombre de *hard reuse* [32]. Anteriormente, en la construcción del tesauro de los accidentes de tránsito, se mostró la reutilización de algunos conceptos del tesauro del SAIJ haciendo únicamente referencia a sus URIs, no se utilizó `owl:import`, este procedimiento es llamado *soft reuse*.

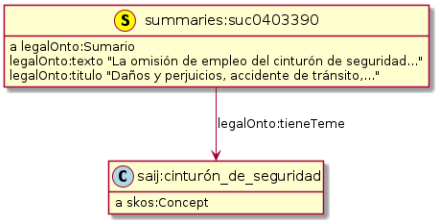
Entonces, los tesauros y la ontología *LegalOnto* son reutilizados en una nueva base de conocimiento, que denominamos *LegalBase*, como módulos ontológicos. Y si bien, existe

[8] Dublin Core es un conjunto estándar de atributos utilizados para describir información bibliográfica.
[9] En versiones previas a la recomendación final de SKOS existía la propiedad `skos:subject`.

Campo	Valor
@ids	http://www.semanticweb.org/summaries#suc0403390
@type	http://www.semanticweb.org/summaries#Sumario
texto	La omisión de empleo del cinturón de seguridad...
título	Daños y perjuicios, accidente de tránsito,...
temas	http://vocabularios.saij.gob.ar/saij/xml.php?skosTema=5259...

Campo	Mapping
texto	http://www.semanticweb.org/legalOnto#texto
titulo	http://www.semanticweb.org/legalOnto#titulo
temas	http://www.semanticweb.org/legalOnto#tieneTema

(a) Un sumario junto a su mapping semántico.



(b) Grafo RDF del sumario.

Figura 5.8: Ejemplo de un sumario transformado al formato RDF.

una superposición entre algunos conceptos SKOS, sus declaraciones se encuentran en distintos espacios de nombres, con lo cual no existe el riesgo de producir inconsistencias cuando los tesauros son importados a *LegalBase*.

Recordemos que tenemos a disposición una colección de sumarios del SAIJ, que se encuentran catalogados con conceptos del tesauro del SAIJ, y para aprovechar esta información en el sistema propuesto debemos incorporarla como fuente de información. Por esta razón, se creo a la ontología *LegalOnto*. Si poblamos[10] a *LegalOnto* con la colección de sumarios catalogados, aumentamos la información sobre las entidades legales, y en consecuencia el rendimiento del sistema de búsqueda propuesto.

Para incluir a los sumarios en la ontología *LegalOnto* necesitamos traducirlos al formato RDF, y esto puede hacerse fácilmente ya que la semi-estructura que brindan los campos de los documentos permiten ver a cada uno de sus campos como a una tripleta —excepto para el campo *@iri*, que indica el IRI del objeto RDF a crear—, solo nos hace falta un *mapping* que asocie a cada campo del documento con la propiedad deseada. En la Figura 5.8a puede verse a un sumario del SAIJ, donde el campo *temas* se obtuvo de aplicar *entity linking* sobre los temas expresados en lenguaje natural con los que fue catalogado. Su posterior transformación al formato RDF se muestra en la Figura 5.8b.

En lugar de incluir a las instancias de los sumarios en la ontología *LegalOnto*, estas se mantienen en un módulo aparte, el cual llamamos *summaries*. Además, el tesauro del SAIJ, originalmente en formato SKOS, es convertido al formato OWL y se le asigna una IRI (`http://vocabularios.saij.gob.ar/saij`) para permitir su importación.

También, se decidió aplicar un razonador OWL sobre la semántica de los tesauros. Para esto fue necesario importar a la ontología que define parte del modelo de datos SKOS[11], ya que los razonadores no incluyen su semántica. Las inferencias se guardan en otro módulo que llamamos *inferred*. Algunas propiedades inferidas —p. ej. `skos:broaderTransitive`— se utilizan en la construcción de las descripciones. Por último, los tesauros y sus inferencias son analizadas con la herramienta Skosify[12], el cual valida y enriquece un vocabulario SKOS acuerdo a las reglas de integridad de la especificación de SKOS y a las buenas prácticas [93].

[10] Poblar una ontología es la tarea de añadir nuevas instancias a la ontología.
[11] `http://www.w3.org/TR/skos-reference/skos-owl1-dl.rdf`.
[12] `https://github.com/NatLibFi/Skosify`

La base de conocimiento *LegalBase* se define de la siguiente manera:

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://www.semanticweb.org/> .

<base> rdf:type owl:Ontology ;
  owl:imports <http://vocabularios.saij.gob.ar/saij> ;
  owl:imports <accidenteDeTránsito> ;
  owl:imports <legalOnto> ;
  owl:imports <inferred> ;
  owl:imports <summaries> .
```

Se crea así una base de conocimiento legal, con la indexación semántica de los sumarios y los Sistema de Organización del Conocimiento con los que fueron indexados, el tesauro del SAIJ y el tesauro de los accidente de tránsito.

5.2. Descripciones de entidades legales

En la Sección 3.1 se describió el proceso de creación de las descripciones de entidades, donde se menciona el uso de un *mapping* o *predicate folding* para la asociación de los predicados de una base de conocimiento con los campos de las descripciones a ser creadas. A continuación, se detalla el *mapping* aplicado a la base de conocimiento legal para la creación de las descripciones de entidades legales.

- *nombres*: contiene el/los nombre/s de la entidad. Los predicados mapeados a este campo son: `rdfs:label`, `skos:prefLabel` y `skos:altLabel`.
- *entidades-relacionadas*: contiene a las entidades que tiene algún tipo de relación con la entidad objetivo, pueden ser entidades más especificas, más generales, etc. Se define para los siguientes predicados: `skos:broader`, `skos:broaderTransitive`, `legalOnto:tieneGeneral`, `skos:narrower`, `skos:related`, `legalOnto:esParteDe`, `skos:narrowerTransitive`, `legalOnto:tieneEspecifico` y por último para el predicado `legalOnto:puedeGenerar`.
- *jurisprudencia-sumarios*: está formado por el texto de todos los sumarios que tienen asignados el tema de la entidad objetivo. Se define mediante el predicado `legalOnto:esTemaEn`, con la resolución dada por el predicado `legalOnto:texto`.
- *jurisprudencia-sumarios-títulos*: comprende a los títulos de los sumarios (palabras claves) que tiene como tema a la entidad objetivo. Se asigna al predicado `legalOnto:esTemaEn`, con la resolución dada por el predicado `legalOnto:titulo`.
- *catch-all*: contiene todos los valores de los demás campos.

Por ejemplo, en la Figura 5.9 se muestra una porción del grafo RDF de la base de conocimiento legal y en la Tabla 5.1 se tiene la descripción de la entidad "cinturón de seguridad", la cual se obtiene aplicando el *mapping* anterior sobre el grafo de la Figura 5.9.

Además de establecer una asociación entre predicados y campos, también es necesario declarar un mecanismo de resolución de URIs. Las URIs no deberían formar parte de las descripciones de entidades, ya que generalmente no contienen términos relevantes para el usuario.

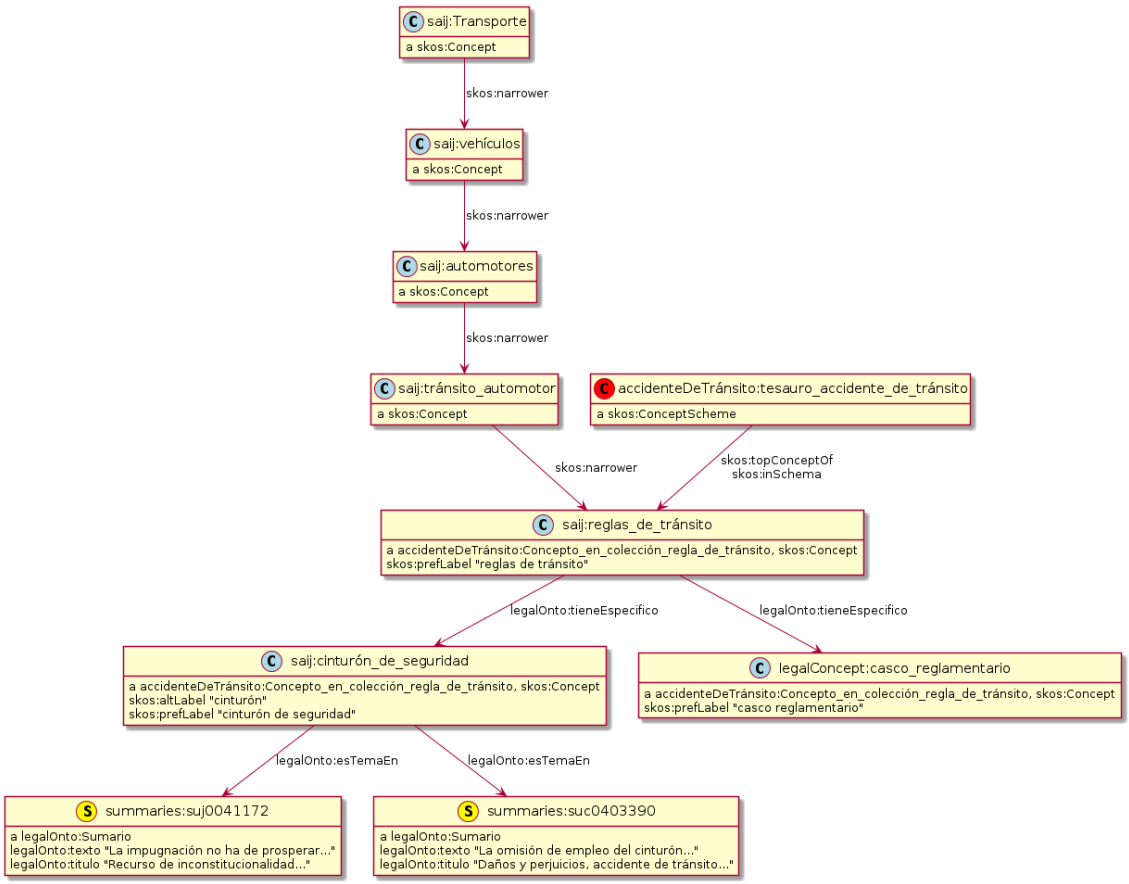


Figura 5.9: Extracto de la ontología de documentos legales.

Campos	**Valor**
nombres	cinturón de seguridad, cinturón.
entidades-relacionadas	reglas de tránsito, tránsito automotor, automotores, vehículos, Transporte, ...
sumarios	La omisión de empleo del cinturón... La impugnación no ha de prosperar...
sumarios-título	Daños y perjuicios, ... Recurso de inconstitucionalidad, ...
catch-all	cinturón de seguridad, cinturón. reglas de tránsito, tránsito automotor, automotores,vehículos, Transporte, ... La omisión de empleo del cinturón... La impugnación no ha de prosperar... Daños y perjuicios, ... Recurso de inconstitucionalidad, ...

Tabla 5.1: Descripción de la entidad “cinturón de seguridad”.

Resolución URIs

Las descripciones de las entidades, como cualquier documento, deberían contener texto concebido para la comprensión de un humano. Las URIs, por otra parte, identifican a un recurso y en general no son compresibles para el humano. Por ejemplo, la URI `http://vocabularios.saij.gob.ar/saij/xml.php?skosTema=5259` identifica al concepto "cinturón de seguridad", no podemos incluir a esta URI en la descripción de la entidad, sino más bien al nombre del concepto que representa. Esto se hizo en el campo *nombres* de la descripción de dicho concepto, Figura 5.1. Entonces, es necesario realizar cierto mecanismo de inferencia para obtener las representaciones textuales de las URIs, esto es llamado *resolución de URIs*. Por ejemplo, en la siguiente tripleta, se muestra una relación de SKOS:

$$(e,\ \texttt{skos:broader},\ e'),$$

esta nos dice que e' es una entidad más general que e. El nombre de e' está contenido en otra tripleta de la base de conocimiento:

$$(e',\ \texttt{rdfs:label},\ e'_{nombre}).$$

Entonces, aplicar la resolución de URIs consiste en reemplazar en la descripción de e la URI e' por su nombre e'_{nombre}, se dice que el nombre de e' está dado por el predicado `rdfs:label`. Siguiendo las agrupaciones de predicados vistas anteriormente, el valor e'_{nombre} formará parte del campo *entidades-relacionadas* de e. Los campos *jurisprudencia-sumario* y *jurisprudencia-sumario-título* utilizan otro predicado, en lugar de `rdfs:label`, para resolver las URIs, estos son los predicados `legalOnto:texto` y `legalOnto:titulo` respectivamente.

5.3. Búsqueda de documentos

Se proponen dos modelos de expansión de la consulta basada en entidades. Estos pueden clasificarse en dos tipos: uno donde el usuario no interviene, realiza una expansión automática, y otro donde sí interviene, una expansión interactiva.

Comenzaremos describiendo el modelo de expansión automática de la consulta, el cual llamamos Modelo de Relevancia con Entidades, luego se describe una versión iterativa de dicho modelo, el Modelo Iterativo de Relevancia con Entidades.

5.3.1. Expansión automática

Cuando se utiliza un algoritmo de retroalimentación por pseudo relevancia como método de expansión de la consulta, se realiza un proceso de generación y selección de documentos relevantes, a partir de los cuales se extraen los términos de expansión de la consulta. Por ejemplo, el algoritmo Modelo de Relevancia genera los documentos candidatos con QL y asume que los documentos con el puntaje más alto son relevantes. En la Figura 5.10, se muestras las etapas de mencionadas cuando se realiza retroalimentación por pseudo relevancia, las cuales estarían incluidas dentro el proceso de "generación y ranqueo de términos" de la Figura 2.1, donde se describe a un algoritmo de expansión de la consulta general.

En el caso de utilizar como fuente de expansión de la consulta, en lugar de los documentos de la colección, a las entidades de una base de conocimiento (o las descripciones de las entidades), se tiene el mismo procedimiento. En la Figura 5.11 se describe las etapas de generación y ranqueo de términos de lo que llamamos retroalimentación por

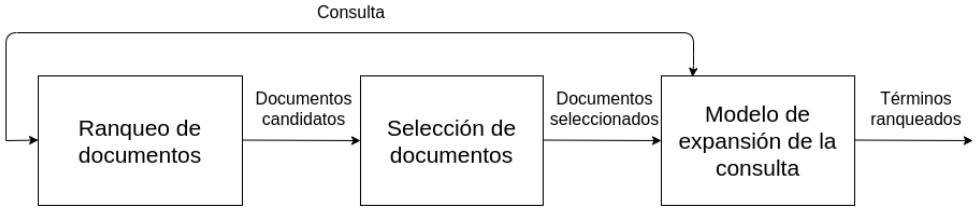


Figura 5.10: Etapas para la generación y ranqueo de términos en un modelo de retroalimentación por pseudo relevancia.

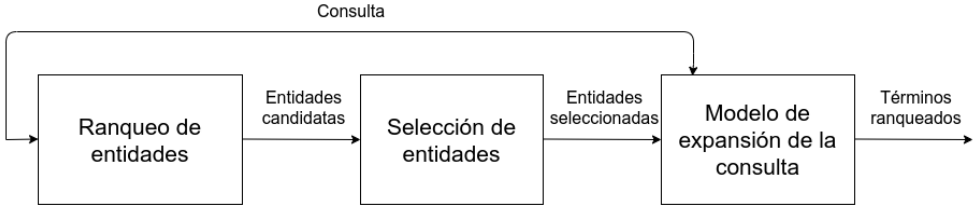


Figura 5.11: Etapas para el ranqueo de términos en un modelo de retroalimentación por pseudo relevancia de entidades.

pseudo relevancia de entidades. Por ejemplo, el Modelo de Lenguaje de Conceptos es un algoritmo de este tipo.

En esta sección, se propone una estimación del Modelo de Lenguaje de Conceptos, presentado en la Sección 3.4, luego se comentan dos extensiones para mejorar su rendimiento.

La expansión de la consulta del Modelo de Lenguaje de Conceptos se obtiene de la siguiente manera:

$$\begin{aligned} P(t|q) &= \frac{P(t,q)}{P(q)} \\ &= \sum_{e \in \mathcal{E}} \frac{P(t,q,e)}{P(q)} \\ &= \sum_{e \in \mathcal{E}} \frac{P(t,e,q)}{P(e,q)} \frac{P(e,q)}{P(q)} \\ &= \sum_{e \in \mathcal{E}} P(t|e,q) P(e|q) \\ &\approx \sum_{e \in \mathcal{E}} P(t|e) P(e|q), \end{aligned} \tag{5.1}$$

donde se asume que la probabilidad de seleccionar un término solo depende del concepto una vez que se ha seleccionada ese concepto para la consulta.

Deben estimarse dos componente, la selección de términos $P(t|e)$ y la selección de entidades $P(e|q)$. Proponemos estimarlas de la siguiente manera, $P(t|e)$ con el modelo del lenguaje de la entidad, $P(t|e) = P(t|\theta_e)$ y $P(e|q)$ con el modelo QL, nosotros asumimos $P(e)$ uniforme, con lo cual se estima con la Ecuación 3.1. En la práctica solo se consideran los términos y las entidades con los puntajes más altos. Es decir, $P(t|e)$ se define tomando los primeros m términos de $P(t|\theta_e)$, lo notamos con el conjunto $\mathcal{V}_q(m)$, y se re-normaliza su puntaje:

$$P(t|e) = \begin{cases} \frac{1}{Z} P(t|\theta_e), & t \in \mathcal{V}_q(m) \\ 0, & sino \end{cases}$$

donde Z es un coeficiente de normalización. Mientras que, $P(e|q)$ toma las primeras k entidades de $score(e, q)$ (Ecuación 3.1), lo notamos con el conjunto $\mathcal{E}_q(k)$, y re-normaliza su puntaje:

$$P(e|q) = \begin{cases} \frac{1}{Z'} score(e, q), & e \in \mathcal{E}_q(k) \\ 0, & sino \end{cases} \tag{5.2}$$

siendo Z' un coeficiente de normalización.

Se puede ver que este modelo es equivalente al Modelo de Relevancia, con entidades en lugar de documentos, es decir, si $P(e|q)$ se estima con el modelo QL (ver ecuación 2.2) y $P(t|e) = P(t|\theta_e)$, como se muestra a continuación:

$$\begin{aligned} P(t|\hat{\theta}_q) &\approx \sum_{e \in \mathcal{E}} P(t|e) P(e|q) \\ &\propto \sum_{e \in \mathcal{E}} P(t|e) P(q|e) P(e) \\ &\approx \sum_{\theta_e \in \Theta} P(\theta_e) P(t|\theta_e) \prod_{i=1}^{n} P(q_i|\theta_e), \end{aligned}$$

se obtiene la misma expansión de la consulta del modelo RM, ver ecuación 2.13. Es por esto que decidimos llamarlo **modelo de relevancia con entidades** (RE).

Luego, la reformulación de la consulta y ranqueo de los documentos es la misma que se utiliza en el trabajo [63]. Es decir, se utiliza la reformulación de la Ecuación 3.10 y luego se utiliza la divergencia KL, de la Ecuación 2.7.

Extensiones

Se detallan dos extensiones que pueden aumentar el rendimiento del modelo propuesto, la incorporación de una semi-estructura en las descripciones de las entidades y una selección de entidades enriquecida, estimada mediante una combinación lineal de puntajes sobre otros aspectos a demás de las términos de la consulta.

En primer lugar, se propone utilizar a los modelos MLM y PRMS para estimar los modelos de lenguaje de las entidades. Si se usa a MLM, entonces tanto $P(t|e)$ como $P(e|q)$ (componentes de la Ecuación 5.1) se estiman en base a la Ecuación 3.3, anotaremos a este modelo como RE+MLM. Mientras que si se utiliza PRMS las componentes anteriores de RE dependen de la Ecuación 3.5, y lo notamos como RE+PRMS. En la experimentación, estos modelos se estiman con la interpolación de Jelinek Mercer.

En segundo lugar, es posible que la selección de entidades dependa no solo de los términos que describen la consulta, sino también de las entidades relacionadas a dicha consulta y a sus tipos. Si tenemos una consulta enriquecida $\tilde{q} = (q, E_q, T_q)$, entonces $score(e, \tilde{q})$ puede ser el resultado de una combinación lineal de los algoritmos de búsqueda vistos en la Sección 3.3, es decir,

$$\begin{aligned} score(e, \tilde{q}) = &\lambda_{term}\ score_{term}(e, q) \\ &+ \lambda_{entity}\ score_{entity}(e, E_q) \\ &+ \lambda_{type}\ score_{type}(e, T_q) \end{aligned} \tag{5.3}$$

donde $score_{term}$ es Query Likelihood (Ecuación 3.1), $score_{entity}$ es la similitud entre entidades (Ecuación 3.7) y $score_{type}$ es la similitud basada en tipos (Ecuación 3.6).

Se propone la siguiente consulta enriquecida:

$$\tilde{q} = (q, entities(q), types(q)), \tag{5.4}$$

siendo $entities(q)$ el resultado de aplicar *entity linking* en la consulta, mientras que $types(q)$ retorna los tipos relacionados a la consulta. En este caso, se estima simplemente como los tipos de $entities(q)$, o sea,

$$types(q) = \{\mathcal{T}_e \mid e \in entities(q)\}.$$

Se han propuesto otras formas no supervisadas más precisas de obtener los tipos relacionados a una consulta [34]. En la experimentación, el método de *entity linking* que se utiliza es el *match* exacto.

Diremos que el modelo RE utiliza un ranking de entidades enriquecido, y lo notamos con un super índice $(*)$, es decir RE*, para diferenciarlo del modelo presentado en la sección anterior. Cuando $\lambda_{term} = 1$ y $\lambda_{entity} = \lambda_{type} = 0$, se tiene el modelo RE original.

Estas dos extensiones podrían combinarse en el modelo RE, lo que resultaría en los modelos RE*+MLM y RE*+PRMS. En total, se propusieron 6 versiones del Modelo de Relevancia con Entidades (RE, RE*, RE+MLM, RE+PRMS, RE*+MLM, RE*+PRMS).

5.3.2. Expansión interactiva

La selección de entidades de la Figura 5.11 puede realizarse de manera interactiva, con la intervención del usuario. Se tendría así un modelo de retroalimentación por relevancia de entidades, donde se le pide al usuario que seleccione entre una lista de entidades sugeridas aquellas que crea relevantes.

Si en el Modelo de Lenguaje de la Consulta, consideramos una selección de entidades interactiva y uniforme, y $P(t|e) = P(t|\theta_e)$, entonces la expansión de la consulta sería:

$$\begin{aligned} P(t|\hat{\theta}_q) &\approx \sum_{e \in \mathcal{E}} P(t|e)P(e|q) \\ &\approx \frac{1}{|\mathcal{E}_q(k)|} \sum_{e \in \mathcal{E}_q} P(t|e) \\ &= \frac{1}{|\mathcal{E}_q(k)|} \sum_{e \in \mathcal{E}_q} P(t|\theta_e) \end{aligned}$$

siendo $\mathcal{E}_q(k)$ el conjunto de las primeras k entidades del modelo QL y que han sido seleccionadas por el usuario, y $P(e|q)$ una probabilidad uniforme en dicho conjunto, es decir,

$$P(e|q) = \begin{cases} c, & e \in \mathcal{E}_q(k) \\ 0, & sino \end{cases}$$

En este modelo de expansión interactiva, primero se genera un conjunto de entidades que se esperan sean relevantes a la consulta mediante QL, llamadas entidades candidatas. En la sección de entidades no se asume, como en el modelo anterior, que las primeras k entidades candidatas sean relevantes. Se le muestran al usuario los resultados y se espera a que elija aquellas que crea relevantes. Luego, se procede a expandir la consulta de acuerdo a las entidades dadas por el usuario

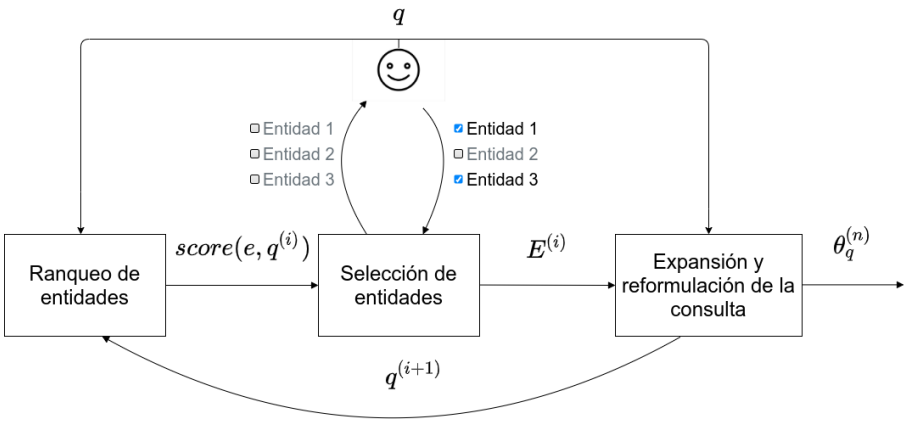


Figura 5.12: Etapas para el ranqueo de términos con retroalimentación por relevancia de entidades.

Este proceso de expansión puede realizarse de manera iterativa, como lo hace el Modelo de Relevancia Iterativo. Sabemos que el modelo IRM muestra mejores rendimiento que RM [16], por lo tanto, pensamos que la expansión interactiva anterior puede mejorar si se realiza de manera iterativa.

Se propone un modelo retroalimentación por relevancia de entidades iterativo. En la Figura 5.12 se describe el proceso de expansión de este modelo. La generación de entidades candidatas, su selección por parte del usuario y la expansión de la consulta se repite n veces. Y en la última iteración, se crea la reformulación de la consulta final, que se anota como $\theta_q^{(n)}$. Pasaremos a detallar el funcionamiento del modelo propuesto.

Las entidades candidatas, aquellas que se muestran al usuario, en la i-ésima iteración, se obtienen de aplicar el modelo KL, es decir:

$$score(e, q^{(i)}) = \sum_{t \in V} P(t|\theta_q^{(i-1)}) \; log \; P(t|\theta_e), \tag{5.5}$$

donde $q^{(i)} = (q, \theta_q^{(i-1)})$, con $1 \leq i \leq n$. Siendo $\theta_q^{(i)}$ la consulta expandida y reformulada de la Ecuación 5.8. Las entidades que ya han sido mostradas en iteraciones anteriores se remueven del ranking.

Las primeras k entidades de la Ecuación 5.5 que son juzgadas como relevantes, en la i-ésima iteración, conforman el conjunto $\mathcal{E}_q^{(i)}(k)$. Estas entidades son añadidas al conjunto de todas las revisiones hechas por el usuario hasta la i-ésima iteración $E_q^{(i)}$, o sea,

$$E_q^{(i)} = E_q^{(i-1)} \cup \; \mathcal{E}_q^{(i)}(k), \tag{5.6}$$

donde $1 \leq i \leq n$, $E^{(0)} = entities(q)$ (*entity linking* sobre la consulta).

Luego, la expansión de la consulta de la i-ésima iteración es la siguiente:

$$P^{(i)}(t|\hat{\theta}_q) = \frac{1}{|E_q^{(i)}|} \sum_{e \in E_q^{(i)}} P(t|\theta_e), \tag{5.7}$$

con $1 \leq i \leq n$. Al igual que el Modelo de Lenguaje de Conceptos, solo se tienen en cuenta a los términos con los puntajes más altos para formar el modelo de la consulta expandida, y con las probabilidades renormalizadas de modo tal de $\sum_t P^{(i)}(t|\hat{\theta}_q) = 1$.

La reformulación de la consulta en la i-ésima iteración es la siguiente:

$$P^{(i)}(t|\theta_q) = \begin{cases} (1-\lambda_q)P^{(0)}(t|\theta_q) + \lambda_q P^{(i)}(t|\hat{\theta}_q) & si \;\; i \geq 1 \\ \frac{c(w,q)}{l_q} & si \; i = 0 \end{cases} \tag{5.8}$$

En la próxima iteración, se ranquean las nuevas entidades candidatas con $\theta_q^{(i)}$, es decir, si $i < n$, entonces:

$$q^{(i+1)} = (q, \theta_q^{(i)})$$

En la última iteración se obtiene la reformulación de la consulta final, $\theta_q^{(n)}$. Luego, esta consulta expandida puede ser utilizada en el modelo KL (Ecuación 2.7).

El modelo propuesto recibe el nombre **modelo iterativo de relevancia con entidades** (IRE), ya que es la versión iterativa del modelo RE, presentado en la sección anterior. Cuando se aplica el modelo de expansión de la consulta IRE y se utiliza para el ranqueo de los documentos al modelo KL, diremos simplemente que utilizamos el modelo IRE. Si IRE se utiliza para la recuperación de entidades, entonces es equivalente al Modelo de Relevancia Iterativo aplicado a la recuperación de entidades.

Un factor que distingue a los modelos propuestos es el tipo de selección de entidades que utilizan en la expansión de la consulta. En el modelo RE, la selección de entidades no es uniforme, el orden de las entidades generadas por el ranking de entidades influye en la expansión de la consulta, es decir, los términos de las entidades más relevantes son más importantes en la expansión de la consulta. En cambio, en el modelo IRE la selección de entidades es uniforme, pues en este modelo se asume que todas las entidades seleccionadas por el usuario son igualmente relevantes.

Extensiones

Como se hizo anteriormente, se proponen las extensiones que incorporan la semi-estructura en las descripciones de las entidades y una selección de entidades enriquecida.

Si se utiliza al modelo MLM, entonces se ve afectada la expansión de la consulta (ver Ecuación 5.7) tanto en la selección de entidades (ver Ecuación 5.5) como en la selección de términos, lo notamos como IRE+MLM. Mientras que si se usa el modelo PRMS, sucede lo mismo, se lo nota como IRE+PRMS.

Si se tiene una consulta enriquecida $\tilde{q} = (q, \theta_q, E_q, T_q)$, se propone el siguiente ranking de entidades enriquecido:

$$\begin{aligned} score(e, \tilde{q}) = {} & \lambda_{term} \; score_{term}(e, \theta_q) \\ & + \lambda_{entity} \; score_{entity}(e, E_q) \\ & + \lambda_{type} \; score_{type}(e, T_q), \end{aligned} \tag{5.9}$$

donde las variables λ_j, con $j \in \{term, entity, type\}$, son los parámetros que definen la importancia de cada puntaje, $score_{term}$ es el modelo KL (Ecuación 2.7), $score_{entity}$ es la similitud entre entidades (Ecuación 3.7) y $score_{type}$ es la similitud basada en tipos (Ecuación 3.6).

Si consideramos la consulta enriquecida de la i-ésima iteración, de la siguiente manera:

$$q^{(i)} = (q, \theta_q^{(i-1)}, E_q^{(i-1)}, types(E_q^{(i-1)})) \tag{5.10}$$

con $1 \leq i \leq n$, entonces podemos utilizar al ranking de entidades enriquecido de la Ecuación 5.9, con $\tilde{q} = q^{(i)}$. Siendo,

$$types(E) = \{\mathcal{T}_e \mid e \in E\}.$$

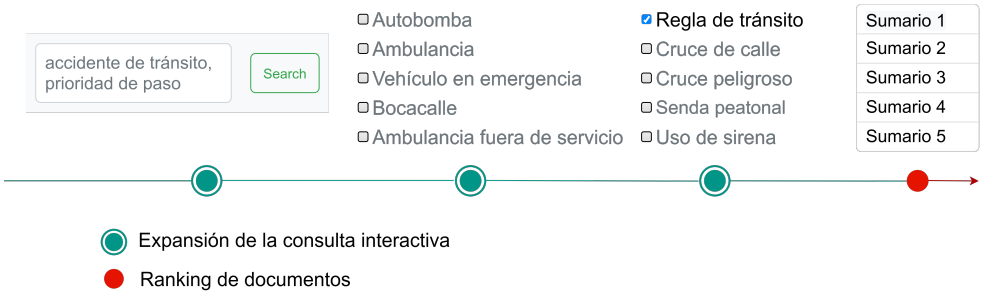


Figura 5.13: Línea de tiempo sobre la búsqueda "accidente de tránsito, prioridad de paso", con expansión de la consulta interactiva.

A diferencia de la consulta enriquecida del Modelo de Relevancia de Entidades, ver Ecuación 5.4, tanto las entidades como los tipos que componen la Ecuación 5.10 dependen la selección del usuario, no solo de las entidades mencionadas en la consulta. Es decir, este ranking de entidades enriquecido tiene más información disponible que su contraparte del modelo RE*. Anotamos a este modelo como IRE$^{(*)}$.

Ejemplo: accidente de tránsito

Si se aplica alguno de los sistemas de búsqueda propuesto sobre la colección de sumarios utilizada en la experimentación y la base de conocimiento legal, ¿cómo está formada la expansión de la consulta para un caso en particular?.

Cuando se utiliza el modelo IRE*, la expansión de la consulta está compuesta por los términos de la consulta original y los términos más importantes tanto de las entidades consideradas relevantes para el usuario como las entidades mencionadas en la consulta original.

Si se utiliza el modelo IRE*+MLM, y se ingresa la consulta "accidente de transito, prioridad de paso", la expansión de la consulta final devuelta por IRE*+MLM tiene al menos los términos asociadas a las entidades "accidente de tránsito" y "prioridad de paso", pertenecientes a *LegalBase*, pues el algoritmo de *entity linking* detecta sus menciones en la consulta. Y si el usuario solo selecciona a la entidad "regla de tránsito", en la segunda y última iteración del algoritmo, donde se muestran 5 entidades por iteración, ver Figura 5.13, entonces la expansión de la consulta final está formada por los términos asociados a estas 3 entidades: "accidente de tránsito", "prioridad de paso", "regla de tránsito". En la Tabla 5.2 pueden verse los términos que forman el modelo de lenguaje de la consulta expandida en cada iteración y los términos de los modelos del lenguajes de las tres entidades mencionadas anteriormente. Los términos se encuentran es su forma raíz, ya que se aplicó *stemming* como procesamiento de las descripciones de las entidades.

Finalmente, a partir de la consulta expandida $\theta_q^{(2)}$ se ranquean los sumarios y son mostrados al usuario.

En resumen, se mostraron dos modelos semánticos de búsqueda, el Modelo de Relevancia con Entidades y el Modelo Iterativo de Relevancia con Entidades. Ambos, utilizan una base de conocimiento para expandir la consulta a partir de los términos asociados a las entidades relacionadas a la consulta, ver Figura 5.14, algo similar a los métodos de retroalimentación por (pseudo) relevancia, los cuales usan a los documentos como medio para expandir la consulta. Pero, a diferencia de los modelos de retroalimentación por relevancia —p. ej., el modelo IRM—, el modelo IRE sugiere entidades ante una consulta, en lugar de documentos, las cuales pueden revisarse por el usuario más rápidamente y

$P(t\|e_1)$	$P(t\|e_2)$	$P(t\|e_3)$	$P^{(0)}(t\|\theta_q)$	$P^{(1)}(t\|\theta_q)$	$P^{(2)}(t\|\theta_q)$
transit	transport	transit	transit	transit	transit
cruc	transit	veloc	accident	accident	pas
derech	automotor	barrer	pas	pas	accident
respons	aere	respons	priorid	transport	priorid
vehicul	accident	pas		derech	transport
automotor	mercad	nivel		cruc	derech
civil	vial	derech		respons	respons
ambul	respons	reglamentari		automotor	cruc
emergent	pasajer	deb		vehicul	automotor
autobomb	carg	conductor		civil	vehicul
				deb	veloc
				emergent	deb
				ambul	barrer
				autobomb	civil
				servici	emergent
					conductor

Tabla 5.2: Modelo del lenguaje de las consultas expandidas en cada iteración del modelo IRE*+MLM y los modelos de las entidades involucradas en la expansión, ante la consulta "accidente de tránsito, prioridad de paso". Se consideraron los primeros 10 términos de $P(t|e_i)$ y los primeros 15 términos de $P^{(i)}(t|\hat{\theta}_q)$. Además, e_1 es "prioridad de paso", e_2 es "accidente de tránsito" y e_3 es "regla de tránsito".

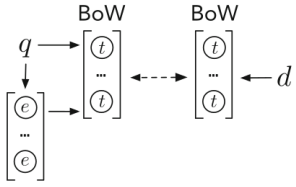


Figura 5.14: Expansión de la consulta basada en entidades. Las entidades relacionadas a la consulta son utilizadas para agregar o re-pesar los términos de la consulta original, lo que resulta en una consulta expandida (Fuente: [9, p. 270]).

cumplen el mismo fin que las recomendaciones de consultas[13], es decir, las sugerencias se relacionan semánticamente con la consulta y permiten precisar la dirección de la búsqueda ya sea para profundizarla o para moverse hacia otros aspectos de la búsqueda. Todos los modelos propuestos junto a sus extensiones son no supervisados, no requieren juicios de relevancia o *query logs*.

Además de los modelos de búsqueda, se vio cómo incorporar a los sumarios, catalogados con conceptos del tesauro del SAIJ, y al tesauro del SAIJ como fuentes de información del sistema propuesto, integrándolos en una sola base de conocimiento. En la construcción de esta base de conocimiento, se describió una manera de implementar a los grupos de conceptos de un tesauro de manera transitiva mediante el uso de clases. Las clases de una base de conocimiento pueden aprovecharse para mejorar la recuperación de entidades, como hacen las extensiones RE* y IRE*.

[13]Las recomendaciones de consultas (*query recommendations*) generalmente explotan las co-ocurrencias y el historial de búsqueda (*query logs*) [9, 302].

Capítulo 6

Experimentación

En este capítulo se evalúa el rendimiento de los modelos de búsqueda propuestos, de acuerdo a una batería de pruebas y se analiza el sistema de recuperación de documentos del SAIJ. En la Sección 6.1 se presentan los conjuntos de test con los que se realiza la evaluación y en la Sección 6.2 se analizan los resultados de los sistemas de recuperación evaluados.

6.1. Colecciones de test

Los sistemas de recuperación de la información tradicionalmente son evaluados a partir de una **colección de test**. Se pueden encontrar colecciones de test diseñadas por campañas de evaluación de sistemas de recuperación de información, o colecciones de test desarrolladas por grupos de investigación con el fin de evaluar un algoritmo de recuperación en un dominio específico [57].

En general, una colección de test se compone de:

- Una colección de documentos.
- Un conjunto de necesidades de información.
- Un conjunto de juicios de relevancia, donde para cada par *consulta-documento* se establece si el documento es relevante o no relevante a la consulta.

Para esta tesina se desarrollaron dos colecciones de test, una para evaluar la recuperación de documentos (los sumarios) y otro para evaluar la recuperación de entidades legales. Si bien nuestro objetivo principal es la recuperación de documentos, también es relevante medir el comportamiento de los algoritmos de recuperación de entidades utilizados en los sistemas de expansión de la consulta propuestos, como se explica luego.

La colección de sumarios *Sumarios20* contiene sumarios del SAIJ, estos se obtuvieron mediante *web crawling* en mayo del 2020. Todos los sumarios pertenecen a la rama del derecho civil y corresponden a la jurisdicción nacional o de la provincia de Santa Fe. En la Figura 6.1 puede verse información estadística de esta colección.

Además, se elaboraron 10 necesidades de información, las cuales fueron diseñadas por un abogado. Todas ellas se refieren a algún tema en particular de la rama del derecho civil, como por ejemplo los accidentes de tránsito. Estas necesidades de información se estructuraron siguiendo el siguiente formato:

- Título: palabras claves que describen la necesidad de información, similar a una consulta que ingresaría un usuario a un motor de búsqueda.

Estadística	Valor
Documentos	45 556
Temas	6 381
Términos	68 674
Tokens	5 212 993
Prom. temas por documento	4.49
Prom. tokens por documento	114.43

Tabla 6.1: Estadísticas de la colección *Sumarios20*.

```
<Num> 8

<Título>
accidente de tránsito, responsabilidad objetiva, cosa riesgosa, embestimiento
de vehículo de menor porte.

<Descripción>
La jurisprudencia, nacional o de la provincia de Santa Fe, relevante debería
discutir acerca de la responsabilidad objetiva o la cosa riesgosa en accidentes
de tránsito que involucren preferentemente embestimientos a vehículos de menor
porte, apoyar la defensa de la víctima, y estar regulada por normas vigentes
hasta el año 2015 inclusive.

<Hechos>
El 25 de Agosto del 2014, el actor circulaba al mando de su moto, haciéndolo a
velocidad reglamentaria y llevando colocado el casco protector. Al arribar a
una intersección, el actor reduce su velocidad de circulación e inicia el cruce
de la misma, haciéndolo detrás de otro rodado que circulaba en idéntico sentido.
Habiendo finalizado el cruce, el actor es embestido desde atrás por un vehículo.
El actor tuvo lesiones, incapacidad, y no trabajó durante su recuperación.
```

Figura 6.1: Ejemplo de una necesidad de información de la colección *Sumarios20*.

- Descripción: Sentencias en lenguaje natural que describen la necesidad de información.
- Hechos: Breve narración de los hechos del caso.

Estos campos se basan en la estructura que siguen las necesidades de información de las colecciones de test de las campañas de recuperación de información TREC, donde se usa "título", "descripción" y "narrativa" para propósitos similares [56, p. 20]. En la Figura 6.1 puede verse a una de las necesidades de información de la colección *Sumarios20*.

Para la creación de los juicios de relevancia es necesario mostrarle al revisor o juez documentos probablemente relevantes para cada una de estas 10 necesidades de información. Estos documentos candidatos reciben el nombre de *pool*. El método que se utilizó para la creación de *pools* se denomina *pooling* de documentos, cada consulta es enviada a diferentes sistemas de búsquedas y los primeros n documentos recuperados por el sistema son añadidos al *pool* de documentos [57].

El *pool* de sumarios fue formado considerando los primeros 20 o 10 documentos que devuelven los modelos a evaluar. Los modelos utilizados generalmente dependen de varios parámetros, se les asignaron a estos parámetros valores arbitrarios, por ejemplo en modelos QL y BM25 se utilizaron los valores por defecto que tienen estos modelos en

```
<Num> 8

<Título>
accidente de tránsito, responsabilidad objetiva, cosa riesgosa, embestimiento
de vehículo de menor porte.

<Descripción>
Se buscan conceptos que se relacionen a la búsqueda de sumarios
número 8 y que podrían utilizarse para realizar otras búsquedas del caso.
```

Figura 6.2: Ejemplo de una necesidad de información de la colección *Entidades20*.

la librería Lucene. Este procedimiento generó un total de 590 documentos candidatos. Luego, cada *pool* fue juzgado por abogados o estudiantes del derecho.

Cabe señalar que la consulta (*query*) no es un sinónimo de la necesidad de información. Una misma necesidad de información puede derivar en diferentes manifestaciones ante distintos sistemas de búsqueda. Por ejemplo, pueden ingresarse pocas palabras claves en un motor de búsqueda de la web y una pregunta en lenguaje natural frente a un asistente de voz [56, p. 20]. Pero, en la práctica se usan indistintamente, es decir, cuando se evalúa un sistema de búsqueda se dice que se hace sobre una consulta, refiriéndose a la necesidad de información.

La consulta ingresada en los sistemas de búsqueda es el texto del campo "título" de la necesidad de información, como suele hacerse en el estado del arte. En general, ingresar la "narrativa", o en este caso "hechos", en el modelo de búsqueda produce peores resultados, ya que suele contener términos que no son importantes y que sirven de distractores para un modelo basado en el *match* exacto de los términos [56, p. 21]. Otra razón por la que se tomó esta decisión es que el motor de búsqueda del SAIJ fue diseñado para buscar sobre palabras claves.

Como se dijo anteriormente, los modelos propuestos dependen de la recuperación de entidades. En particular, si se quiere evaluar el modelo de expansión interactiva es necesario simular la interacción que realiza el usuario en el proceso de expansión de la consulta. Es decir, deben conocerse cuáles son los conceptos que más se relacionan a las necesidades de información planteadas en la colección de test de sumarios. Entonces, debe crearse otra colección de test para evaluar la recuperación de entidades, construida sobre las mismas necesidades de información de *Sumarios20*.

La evaluación de un sistema de recuperación de entidades es análoga a un sistema de recuperación de documentos, cada entidad se presenta como un documento. La colección de test utilizada para la evaluación de la recuperación de entidades legales, que llamaremos *Entidades20*, se compone de la colección de descripciones de entidades creadas a partir de la base de conocimiento legal presentada en la Sección 5.1.

Las necesidades de información de la colección *Entidades20* son similares a las necesidades de información de *Sumarios20*, pero se preguntan por entidades en lugar de documentos. En la Figura 6.2 se muestra la necesidad de información asociada a la búsqueda sobre accidentes de tránsito de la Figura 6.1

Los juicios de relevancia sobre las entidades fueron realizados por las mismas personas que revisaron los documentos. En total se juzgaron 450 conceptos legales.

En la siguiente sección, se describe la evaluación de los algoritmos de búsqueda propuestos y otros modelos que han sido presentados en esta tesina.

6.2. Evaluación

Las métricas que cuantifican la calidad de un modelo de búsqueda se computan a partir de los juicios de relevancia. Estos pueden definirse como un conjunto de tripletas (q, d, r), donde es r una variable binaria que indica si el documento d es o no es relevante a la consulta q. También, pueden usarse juicios no binarios, lo que se denomina juicios de relevancia por grados. Se define la relevancia de un documento como la función $rel(q, d) \in \{0, 1\}$. A continuación, se detallan las métricas con las que se evalúan a los algoritmos de búsqueda, siguiendo la notación dada en [56]. Se considera una lista de ranqueo de la forma $R = \{(i, d_i)\}_{i=1}^{l}$, siendo l su longitud. En general, las métricas se computan hasta una cierta longitud, es decir $\{(i, d_i)\}_{i=1}^{k}$, con $k \leq l$, lo cual se denota como Métrica@k.

La **precisión** (*precision*) se define como la fracción de documentos en una lista de ranqueo R que son relevantes:

$$Precision(R, q) = \frac{\sum_{(i,d)\in R} rel(q, d)}{|R|},$$

donde $rel(q, d)$ indica si el documento d es relevante a la consulta q, asumiendo relevancia binaria.

La **exhaustividad** (*recall*) es definida como la fracción de documentos relevantes (en toda la colección $\mathcal{C}$) que son retornados en la lista ranqueada R:

$$Recall(R, q) = \frac{\sum_{(i,d)\in R} rel(q, d)}{\sum_{d\in\mathcal{C}} rel(q, d)}.$$

Estas métricas tiene la ventaja que de son fáciles de interpretar, pero no tienen en cuenta el orden en el que aparecen los documentos relevantes.

La **precisión promedio** (*average precision*) se define como sigue:

$$AP(R, q) = \frac{\sum_{(i,d)\in R} Precision@i(R, q)\ rel(q, d)}{\sum_{d\in\mathcal{C}} rel(q, d)}.$$

La precisión promedio realiza el promedio de las precisiones que se obtienen hasta las posiciones donde se encuentran los documentos relevantes en el ranking R, siendo la contribución de documentos no relevantes 0. Esta métrica debería calcularse sobre una lista de ranqueo que incluya a todos documentos relevantes, por ello no es conveniente truncarla en las primeras posiciones, en la práctica suele utilizarse como corte los primeros 1 000 documentos del ranqueo. Este método captura aspectos tanto de precisión como exhaustividad y se favorece de la aparición de documentos relevantes en los primeros lugares de la lista de ranqueo.

Estas tres medidas resumen la efectividad del sistema para una sola consulta. Pero, para reflejar el rendimiento de un algoritmo de recuperación es necesario evaluar su efectividad sobre todo el conjunto de consultas. Una manera sencilla de mostrar la efectividad sobre múltiples consultas es el promedio de los valores que se obtienen con cada métrica vista. En el caso de la precisión promedio, su promedio sobre todas las consultas recibe el nombre de **media de la precisión promedio** (*mean average precision*). Se la define de la siguiente manera:

$$MAP = \frac{1}{|Q|} \sum_{(R,q)\in runs} AV(R, q),$$

siendo $runs$ el conjunto de listas de rankings producidas por el sistema para cada consulta en Q. A las demás métricas, es decir precisión y exhaustividad, cuando son promediadas se las anota de la misma forma.

Además, cuando se evalúan algoritmos búsqueda con expansión de la consulta es importante considerar la robustez del sistema. Los algoritmos de expansión de la consulta tienen una marcada variabilidad en las consultas, mientras que la mayoría son mejoradas, algunas son perjudicadas [21, p. 34]. Una medida que usualmente se utiliza para medir la robustez del sistema es el **índice de robustez** (*robustness index, RI*), se define como la porción de la diferencia entre el número de consultas mejoradas y aquellas perjudicadas por el algoritmo de expansión de la consulta, con respecto a los resultados de un algoritmo de búsqueda sin expansión de la consulta.

$$RI = \frac{N_+ - N_-}{|Q|},$$

donde $|Q|$ denota el número de consultas, y N_+/N_- indican el número de consultas mejoradas o perjudicadas en comparación al rendimiento de un algoritmo sin expansión de la consulta. Si R_1 es la lista ranqueada del algoritmo con expansión de la consulta y R_2 es el ranking del algoritmo sin expansión, entonces N_+ y N_- se pueden definir de la siguiente manera:

$$N_+ = |\{q \in Q \mid AV(R_1, q) > AV(R_2, q)\}|$$
$$N_- = |\{q \in Q \mid AV(R_1, q) < AV(R_2, q)\}|$$

Aquí, se consideran ganancias o perdidas mayores al 10 %, para evitar la influencia de pequeñas diferencias en la precisión promedio.

Un detalle que es crucial en la interpretación de estas métricas son los documentos no juzgados, ¿qué ocurre si la lista de ranking R contiene documentos que no se encuentran en los juicios de relevancias correspondiente a la consulta?, lo cual sucede a menudo. El tratamiento estándar es considerar a los documentos no juzgados como irrelevantes. Entonces, la evaluación de un algoritmo puede ser pobre no por que el algoritmo sea deficiente, si no por que el algoritmo produce muchos resultados que no fueron juzgados. Una manera de diagnosticar este potencial problema es computar la fracción de documentos juzgados con un corte k, que se nota $Judged@k$ o $J@k$. Por ejemplo, si el 80 % de los primeros 10 resultados no fue juzgado, encontraríamos, a los sumo, $P@10 = 20$.

Los valores absolutos de una métrica de evaluación deben interpretarse cuidadosamente, ya que todos los juicios de relevancia son opiniones [56, p. 22]. De todas manera, es posible realizar comparaciones entre los sistemas de búsqueda que sean confiables si se tiene una colección de test bien construida. En este caso, los pocos juicios de relevancia en las colecciones de test *Sumarios20* y *Entidades20* no permiten concluir la superior de un sistema frente al otro. Se recomienda contar con al menos 30 juicios de relevancia para mostrar estadísticamente la superioridad de un sistema de búsqueda [74]. Pese a esto, encontramos resultados que nos permiten entender el funcionamiento de los algoritmos propuestos.

Por último, se describe el proceso de normalización aplicado en la indexación de los documentos y las consultas:

1. Reducción a minúsculas.
2. En la tokenización se consideró como *token* a 2 o más caracteres alfanuméricos, descartando los signos de puntuación. Se utilizó la siguiente expresión regular (*regex*) para generar los *tokens*: `(?u)\b\w\w+\b`[1].
3. Remoción de palabras vacías de acuerdo a la lista en español de Snowball[2].

[1] Es un *regex* estándar para la extracción de *features* de la librería scikit-learn.
[2] `https://snowballstem.org/algorithms/spanish/stop.txt`

Parámetro		Descripción
λ	Ec. 3.2	Interpolación para el suavizado Jelinek-Mercer.
λ_f	Ec. 3.4	Interpolación para el suavizado Jelinek-Mercer del campo f, tanto para el modelo MLM como PRMS.
α_f	Ec. 3.3	Peso del campo f en el modelo MLM.
λ_q	Ec. 2.17	Interpolación entre la consulta inicial y la consulta expandida.
terms	Ec. 2.16	Número de términos a considerar en la consulta expandida.
documents	Ec. 2.18	Número de documentos a recuperar por iteración. En este caso, los documentos son entidades.
iterations	Ec. 2.15	Número de iteraciones.

Tabla 6.2: Parámetros libres de los modelos QLJM, MLM, PRMS y IRM aplicados en la recuperación de entidades.

4. Utilización del *stemmer* de Porter en su versión español[3].

Seguidamente, la evaluación de los algoritmos de búsqueda se organizan en dos secciones: la recuperación de entidades legales y la recuperación de los sumarios. Analizar los algoritmos de búsqueda de entidades será útil para entender el funcionamiento de los algoritmos de expansión de la consulta propuesto, que se basan en la recuperación de entidades. A medida que se desarrolla la experimentación se plantean las preguntas que motivaron las pruebas.

Recuperación de entidades legales

En esta sección, se analiza el rendimiento de los algoritmos de búsquedas basados en modelos de lenguaje aplicados a la recuperación de entidades. El objetivo de esta tesina es mejorar la recuperación de documentos, pero también queremos saber si mejorando la recuperación de entidades mejoramos la recuperación de documentos para los algoritmos propuestos en esta tesina, esto se verá más adelante. En primer lugar, se realiza la evaluación de modelos de búsqueda que no expanden la consulta, estos son: QJLM, MLM y PRMS. Luego, se evalúa el rendimiento de los siguientes modelos con expansión interactiva de la consulta: IRM, IRM+MLM, IRM+PRMS.

La evaluación de estos algoritmos de búsqueda se realiza mediante *4-fold cross-validation* y sus parámetros se optimizan mediante *grid search*. El método *grid search* consiste en aplicar fuerza bruta sobre un espacio de parámetros del algoritmo de búsqueda de modo tal de encontrar los valores que maximicen alguna métrica. En la recuperación de información, en general se maximiza el MAP. A causa de la complejidad exponencial de *grid search* y el gran número de parámetros que tienen los modelos a evaluar se decide optimizar solo a algunos de ellos. En la Tabla 6.2 se muestran los parámetros de los modelos evaluados.

Los parámetros optimizados son: la interpolación del suavizado de Jelinek Mercer (λ, λ_f) y los pesos de los campos de las descripciones (α_f), con valores en el intervalo $[0, 1]$ y saltos de 0.25. La elección de estos parámetros, tanto en esta sección como en las secciones posteriores, se debe a su gran influencia en el rendimiento del sistema de búsqueda, además son comunes a todos los algoritmos basados en modelos del lenguaje con los que se experimenta. Al resto de los parámetros se les fija un valor arbitrario estándar, los valores son: $\lambda_q = 0.25, terms = 15, documents = 10$, y $iterations = 2$, estos parámetros se utilizan en el modelo IRM. Entonces, el modelo IRM aplicado a la recuperación de entidades, considera a lo sumo 15 términos para expandir la consulta inicial y muestra

[3] `https://snowballstem.org/algorithms/spanish/stemmer.html`

Colección	Métrica	QLJM	MLM	PRMS
Entidades20	MAP	0.339	0.378	0.327
	P@10	0.390	0.410	0.360
	P@20	0.275	0.285	0.280
	J@20	0.909	0.885	0.890

Tabla 6.3: Resultados de la evaluación de sistemas de recuperación de entidades sin expansión de la consulta.

f	t =*"artículo"* $P(f\|t)$	t =*"casco"* $P(f\|t)$	t =*"recurso"* $P(f\|t)$	t =*"tránsito"* $P(f\|t)$
`nombres`	0.010	0.476	0.077	0.076
`entidades-relacionadas`	0.004	0.400	0.096	0.290
`jurisprudencia-sumario`	0.533	0.031	0.057	0.039
`jurisprudencia-sumario-titulo`	0	0.048	0.634	0.489
`catch-all`	0.451	0.042	0.133	0.103

Tabla 6.4: Ejemplo del mapping de probabilidades del modelo PRMS.

al usuario un total de 20 entidades. Pero, el número de parámetros sigue siendo alto para ser optimizados en un ordenador doméstico —p. ej, el modelo IRM+MLM tiene 14 parámetros—, por lo que también se ignoran ciertos campos de las descripciones en algunos modelos. Más precisamente, en los modelos IRM+MLM y IRM+PRMS se omiten los campos `nombres` y `jurisprudencia-sumario-título`, asignándoles pesos nulos.

La primera pregunta que nos hacemos es la siguiente:

- ¿La semi-estructura de las entidades (MLM ó PRMS) favorece la recuperación de entidades?.

En la Tabla 6.3 se muestran los resultados de los modelos que no realizan expansiones de la consulta. El mejor modelo en términos de MAP es MLM. El uso de campos en las descripciones mejora el rendimiento de la recuperación de entidades, pero solo si se utiliza el modelo MLM. Si se emplea el modelo PRMS se alcanza un rendimiento similar en términos de MAP al modelo QLJM, el cual no tiene en cuenta la semi-estructura de las descripciones de entidades. Sabemos que el modelo PRMS logra un buen funcionamiento si los campos de los documentos —en este caso las descripciones de las entidades— tienen distribuciones de términos distintivas y esto parece cumplirse en la colección analizada. En la Tabla 6.4 se muestran los pesos dinámicos que asigna PRMS para algunos términos. En este ejemplo, los campos `nombres` y `entidades-relacionadas` favorecen a las palabras que son conceptos del tesauro del SAIJ (casco, tránsito), el campo `jurisprudencia-sumario-titulo` favorece a los términos que son palabras claves de un sumario (tránsito, recurso), mientras que los campos `jurisprudencia-sumario` y `catch-all` favorecen al resto de los términos legales (artículo).

De acuerdo a estos resultados, los campos `jurisprudencia-sumario` y `catch-all` podrían recibir pesos bajos en el modelo PRMS cuando las consultas son palabras claves, como sucede en esta colección de test. Sin embargo, estos campos son importantes para el ranqueo, pues contienen la mayor cantidad de información de una entidad. El modelo MLM optimizado les asigna a estos campos un peso mayor a 0.25. Por lo tanto, la falta de uniformidad en la cantidad de información de cada campo de las descripciones, como en la colección *Entidades20*, perjudicaría al modelo PRMS En el caso de las búsquedas sin expansión de la consulta, el modelo MLM muestra un mejor rendimiento que PRMS.

Luego de analizar los algoritmos sin expansión de la consulta, nos preguntamos:

Colección	Métrica	IRM	IRM+MLM	IRM+PRMS
Entidades20	MAP	0.353	0.391	0.349
	P@10	0.390	0.440	0.380
	P@20	0.285	0.325	0.285
	J@20	0.905	0.890	0.850

Tabla 6.5: Resultados de la evaluación de sistemas de expansión de la consulta interactiva para la recuperación de entidades.

- ¿Cuál modelo que incorpora la semi-estructura de las entidades (MLM ó PRMS) mejora la recuperación de entidades de manera iterativa?.

A continuación, se evalúan los algoritmos de búsqueda IRM, IRM+MLM y IRM+PRMS, siguiendo el mismo procedimiento anterior, *4-fold cross validation*, *grid search* y el espacio de parámetros ya mencionado.

Es importante señalar que la comparación de los modelos de expansión de la consulta, basados en la retroalimentación de relevancia, no debería realizarse sobre los resultados que arrojan en la última iteración. Si se evalúa un sistema de búsqueda sobre los mismos documentos que revisó el usuario se obtendrían ganancias en términos de MAP superiores al 50 % [58, p. 171]. Una evaluación como la anterior no sería justa, ya que los documentos marcados como relevantes son favorecidos en el ranqueo. Solo deberíamos evaluar a un algoritmo con respecto a documentos no vistos por el usuario. Por esta razón, se utiliza el método denominado *frezzing ranking*, el cual define una lista de resultados "totales" formada por los resultados devueltos en cada iteración del algoritmo. Esta lista se construye de acuerdo al orden con que se muestran los resultados, documentos mostrados en iteraciones anteriores se remueven de la lista. Los resultados de la última iteración son concatenados al final de esta lista de resultados totales [16, 15]. Además, la selección de las entidades relevantes es simulada a partir de los juicios de relevancia.

En la Tabla 6.5, se muestran los resultados de las evaluaciones de los modelos iterativos. Cada modelo iterativo muestra más resultados relevantes que su contraparte sin expansión interactiva de la consulta, es decir el modelo IRM+MLM es superior en términos de MAP al modelo MLM, sucede lo mismo entre IRM+PRMS y PRMS, ver Tabla 6.3. El modelo IRM+MLM muestra los mejores resultados en términos de MAP.

Por lo tanto, en la colección *Entidades20* el ajuste estático (MLM) de los pesos asociados a los campos de las descripciones de entidades alcanza un mayor rendimiento frente al ajuste dinámico (PRMS), tanto en búsquedas sin expansión de la consulta como con expansión iterativa.

Recuperación de sumarios

En esta sección se analiza la búsqueda de sumarios y su relación con la búsqueda de entidades. En primer lugar, se evalúan los algoritmos de expansión de la consulta RE, IRE, IRE* y RE*, sobre la colección *Sumarios20*. Luego, se evalúan los modelos IRM y IRM* (se lo define más adelante) en la colección *Entidades20*.

La pregunta que nos hacemos es la siguiente:

- ¿La recuperación de sumarios mejora si la selección de entidades incorpora información semántica?.

Cuando se utilizan los algoritmos de búsqueda basados en el Modelo de Lenguaje de Conceptos, la selección de entidades expresa cuales son las entidades relevantes para la consulta. Esta selección se estima a partir de un ranking de entidades. Cuando el

ranking incorpora información semántica, como en los modelos RE* y IRE*, donde no solo se consideran a los términos de la consulta sino también a las entidades mencionadas en ella y a sus tipos, decimos que tenemos una selección de entidades semántica, pues utiliza un *ranking de entidades enriquecido* con una consulta aumentada con entidades y tipos. Cuando el ranking no incorpora dicha información decimos que es un *ranking de entidades basado en términos*, como ocurre en los modelos RE y IRE.

Por otra parte, se define al modelo IRM* como el modelo IRM. En este caso, es aplicado a la recuperación de entidades, donde la Ecuación 2.18 es reemplazada por la Ecuación 5.9 con $RP^{(i)} = E^{(i)}$. Es decir, el modelo IRM* recupera entidades con una selección de entidades semántica. Así, los modelos IRE* y IRM* son prácticamente iguales, pues ante una consulta producen la misma expansión, pero la expansión de la consulta producida por el modelo IRM* se aplica a la recuperación de entidades, mientras que la expansión del modelo IRE* se utiliza en la recuperación de documentos.

Tanto en la colección *Sumario20* como *Entidades20*, la evaluación se lleva a cabo sin dividir a los datos en un conjunto de entrenamiento y en otro de prueba, a diferencia de lo que se hizo anteriormente, es decir, la optimización de los parámetros y la posterior evaluación se realiza sobre el mismo conjunto de datos. La aplicación de esta metodología dista de ser justa si se pretende comparar los algoritmos y generalizar su rendimiento. Sin embargo, este no es nuestro objetivo, pretendemos hallar la cota superior en los rendimientos de los modelos evaluados, sobre las colecciones *Sumarios20* y *Entidades20*, como se hace en [63]. La división de los datos en dos conjuntos sería innecesaria si las consultas contenidas en los juicios de relevancia fuesen las únicas consultas que se ingresan en el sistema [27, p. 331], quizás esto resulte apropiado si se tuviesen consultas representativas y un dominio de aplicación acotado.

Los parámetros de los modelos RE y IRE se muestran en la Tabla 6.6, los parámetros del modelo IRM se encuentran en la Tabla 6.2 y con respecto al modelo IRM*, este tiene todos los parámetros del model IRM, más las variables de interpolación del ranking de entidades enriquecido. Solo se optimizaron algunos parámetros, estos son: la variable de suavizado Jelinek-Mercer (λ_e) en los modelos del lenguaje de entidades, con valores en el intervalo $[0, 1]$ y saltos de 0.25, las variables del ranking de entidades enriquecido ($\lambda_{term},\ \lambda_{entity},\ \lambda_{type}$), con valores en el intervalo $[0, 1]$ y saltos de 0.05. El resto de los parámetros toman los siguientes valores: $\lambda_d = 0.75,\ \lambda_q = 0.25$ (IRE, IRM), $\lambda_q = 0.5$ (RE), $terms = 15,\ documents = 10, entities = 10,\ entities\text{-}rel = 10$, y $iterations = 2$. Los distintos valores que toma λ_q se debe a que se observan mayores rendimientos en la expansión automática (RE) con una interpolación superior a 0.25, mientras en la expansión interactiva (IRE, IRM) los mayores rendimiento ocurren con una interpolación menor a 0.25.

En la Tabla 6.7 puede verse que el uso del ranking de entidades enriquecido mejora la recuperación de sumarios, pero solo cuando se utiliza la expansión automática de la consulta (RE*) y no cuando se realiza la expansión iterativa (IRE*). Entonces, en la colección *Sumarios20*, las expansiones de la consulta del modelo IRE*, las cuales se obtienen de utilizar el ranking de entidades enriquecido, no superan las expansiones generadas por el modelo IRE que utiliza un ranking de entidades basado en términos.

En principio, la expansión de la consulta a partir de la retroalimentación por pseudo relevancia —es decir añadir términos a la consulta a partir de documentos que se obtienen de manera automática y que se suponen son relevantes a dicha consulta— tendría un rendimiento inferior a la retroalimentación por relevancia —expandir la consulta con los términos pertenecientes a los documentos seleccionados por el usuario—. Sin embargo, cuando se aplican estos mecanismos de retroalimentación sobre las descripciones de las entidades, la ventaja anterior ya no se cumple para la colección *Sumarios20*. Suponemos

Parámetro		Descripción
λ_d	Ec. 2.5	Interpolación del suavizado Jelinek-Mercer para modelos del lenguaje sobre documentos.
λ_e	Ec. 3.2	Interpolación del suavizado Jelinek-Mercer para modelos de lenguaje sobre entidades.
λ_q	Ec. 5.8, Ec. 3.10	Interpolación entre la consulta inicial y la consulta expandida.
$\lambda_{term}, \lambda_{entity}$ λ_{type}	Ec. 5.3, Ec. 5.9	Interpolación del ranking de entidades enriquecido.
terms	Ec. 5.7, Ec. 3.8	Número de términos a considerar en la consulta expandida.
entities	Ec. 5.6	Número de entidades sugeridas por iteración (IRE).
entities-rel	Ec. 5.2	Número de entidades consideradas relevantes (RE).
iterations	Ec. 5.6	Número de iteraciones.

Tabla 6.6: Parámetros libres de los modelos RE y IRE, considerando que los modelos de lenguaje de los documentos utilizan el suavizado Jelinek-Mercer.

Colección	Métrica	RE	RE*	IRE	IRE*	$\text{IRE}^*_{\text{IRM}^*}$
Sumarios20	MAP	0.331	0.372	0.317	0.317	0.313
	P@10	0.250	0.310	0.290	0.290	0.270
	P@20	0.200	0.205	0.225	0.225	0.210
	J@20	0.820	0.890	0.975	0.934	0.985

Tabla 6.7: Comparación de los sistemas de expansión iterativa de la consulta.

que los términos asociados a una entidad relevante podrían no ser tan buenos como los términos de un documento relevante. Las entidades que mejorarían la consulta en el modelo IRE no serían necesariamente las entidades que entiende el usuario como relevantes, existiría cierta dicotomía entre las entidades que el usuario considera como relevantes y aquellas entidades que serían útiles para expandir la consulta. Pero, quizás esto se deba a características propias de la base de conocimiento legal, descripciones de entidades de mayor calidad —como las entidades de DBpedia— podrían proveer mejores términos para expandir la consulta.

Entonces, la falta de utilidad de las entidades que el usuario considera relevantes explicaría la diferencia entre los modelos RE y IRE. El rendimiento del modelo RE es superior al modelo IRE, ya que la selección automática de entidades no se acota a las entidades que el usuario considera como relevantes y esta diferencia es mayor cuando se utiliza el ranking de entidades enriquecido. Si bien la selección de entidades semántica puede mejorar la recuperación de sumarios, como muestra el modelo RE*, la expansión iterativa de la consulta no puede aprovecharlo si el usuario no considera como relevantes a aquellas entidades candidatas que podrían mejorar la consulta inicial y por ello el modelo IRE* mantiene el mismo rendimiento de IRE.

La siguiente pregunta que nos hacemos es:

- ¿Si en la expansión iterativa se sugiere al usuario una gran cantidad de entidades relevantes, la recuperación de sumarios mejora?.

El modelo IRM* optimizado en la colección *Entidades20* le muestra al usuario un gran número de entidades relevantes, ver Tabla 6.8. Si queremos conocer el rendimiento del modelo IRE* cuando sugiere la mayor cantidad de entidades relevantes, equivale a evaluar el modelo IRE* con los valores de los parámetros del modelo IRM* optimizado en *Entidades20*, lo anotamos como $\text{IRE}^*_{\text{IRM}^*}$.

En la Tabla 6.7 se puede ver que la mejor expansión de la consulta para la recuperación de entidades no es la mejor para la recuperación de sumarios. Por lo tanto, mostrarle

Colección	Métrica	IRM	IRM*
Entidades20	MAP	0.365	0.388
	P@10	0.390	0.420
	P@20	0.300	0.325
	J@20	0.909	0.845

Tabla 6.8: Comparación de sistemas de expansión interactiva de la consulta para la recuperación de entidades.

	IRE	IRE*	$\text{IRE}^*_{\text{IRM}^*}$
Entidades seleccionadas	6	3.7	6.5
Tamaño de la consulta	19.4	19.4	19.8

Tabla 6.9: Características de distintos tipos de búsqueda en el modelo IRE.

al usuario más entidades relevantes no implica una mejor reformulación de la consulta, al menos para la recuperación de sumarios. En la Tabla 6.9 se muestra la cantidad de términos que forman la consulta expandida[4] y el número de entidades seleccionadas por los modelos de expansión interactivos.

En suma, en la colección *Sumarios20*, solo el modelo de expansión automática parece mejorar su rendimiento cuando se utiliza una selección de entidades semántica. Además, el mejor rendimiento del algoritmo de expansión iterativa de la consulta no se alcanza cuando sugiere al usuario la mayor cantidad de entidades relevantes.

Búsqueda con selección de entidades basada en términos

Se analizan, sobre la colección *Sumario20*, los siguientes modelos de expansión de la consulta centrada en entidades RE, RE+MLM, RE+PRMS, IRE, IRE+MLM, IRE+PRMS, junto con el algoritmo RM3 y los otros modelos que no realizan expansión de la consulta, estos son QLD, QLJM, BM25, TF-IDF.

La evaluación de todos estos algoritmos de búsqueda se realiza mediante *4-fold cross-validation*, pues evita sobrestimar los resultados para una colección en particular. Los parámetros se ajustan con *grid search*. Los parámetros de los modelos evaluados están en la Tabla 6.6, Tabla 6.10 y Tabla 6.2 —de esta última tabla se consideraron los parámetros correspondientes a los modelos MLM y PRMS—.

Los parámetros optimizados son: la interpolación del suavizado de Jelinek-Mercer ($\lambda,\ \lambda_e,\ \lambda_f$) y los pesos de los campos de las descripciones (α_f), para los algoritmos basados en modelos del lenguaje, con valores en el intervalo $[0, 1]$ y saltos de 0.25. A los demás parámetros se les asignaron los siguiente valores: $\mu = 2000,\ \lambda = 0.7,\ k_1 = 1.2,\ b = 0.75,\ \lambda_d = 0.75,\ \lambda_q = 0.25$ (RM3, IRE, IRE+MLM, IRE+PRMS), $\lambda_q = 0.5$ (RE, RE+MLM, RE+PRMS), $terms = 15$, $documents = 10$, $entities = 10$, $entities\text{-}rel = 10$, $iterations = 2$. Se ignoraron los campos `nombres` y `jurisprudencia-sumario-titulo` en los modelos RE+MLM y RE+PRMS. Además, en los modelos IRE+MLM y IRE+PRMS se omitió el campo `jurisprudencia-sumario-titulo`.

Las preguntas que motivaron la evaluación de esta sección son las siguientes:

- ¿Los modelos propuestos superan el rendimiento de algoritmos que no realizan expansiones de la consulta?.
- ¿Entre las extensiones que utilizan a los modelos MLM y PRMS, cuál produce los mejores resultados?

[4]El tamaño de una consulta q es $|\{t \in \mathcal{V} \mid P(t|\theta_q) > 0\}|$.

Parámetro		Descripción
μ	Ec. 2.6	Parámetro del suavizado de Dirichlet.
λ	Ec. 3.2	Interpolación para el suavizado Jelinek-Mercer.
k_1, b	Ec. 1.2	Parámetros del modelo BM25.
λ_q	Ec. 2.14	Interpolación entre la consulta inicial y la consulta expandida.
terms	Ec. 2.13	Número de términos a considerar en la consulta expandida.
documents	Ec. 2.13	Número de documentos a considerar como relevantes.

Tabla 6.10: Parámetros libres de los modelos QLD, QLJM, BM25, RM3.

Colección	Métrica	QLD	QLJM	BM25	TF-IDF
Sumarios20	MAP	0.187	0.284	0.282	**0.320**
	P@10	0.150	**0.250**	0.210	**0.250**
	P@20	0.160	0.190	0.195	**0.199**
	J@20	0.994	**1.000**	**1.000**	**1.000**

Tabla 6.11: Resultados de la evaluación de sistemas de recuperación de documentos sin expansión de la consulta.

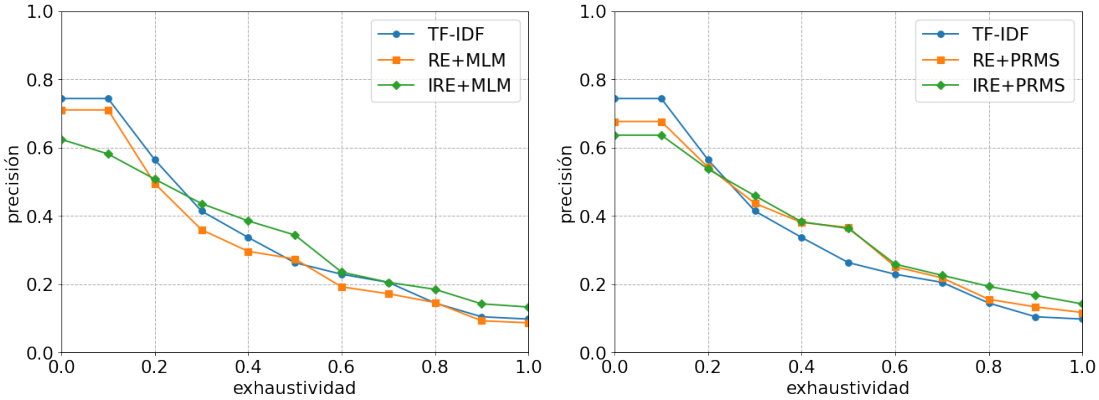
- ¿La expansión interactiva de la consulta mejora la expansión automática?.

En la Tabla 6.11 se muestran los resultados que se obtuvieron de evaluar los sistema de recuperación de documentos sin expansión de la consulta. El mejor resultado, en términos de MAP, lo obtiene el modelo TF-IDF, este el único de los todos los algoritmos evaluados que no tiene parámetros. El resto de modelos, es decir QLD, QLJM y BM25, tienen valores en sus parámetros por defectos, no fueron optimizados y por esta razón sus rendimientos no superarían a TF-IDF.

Por otro lado, en la Tabla 6.12 se comparan los modelos con expansión de la consulta. Los modelos que utilizan a PRMS, es decir los modelos RE+PRMS y IRE+PRMS, alcanzan los mayores rendimientos en términos de MAP, superiores a TF-IDF. Para más detalle, puede verse en la Figura 6.3b la curva de precisión y exhaustividad[5] de dichos modelos.

El uso del modelo MLM alcanza rendimientos inferiores al modelo PRMS, pues los modelos RE+MLM, IRE+MLM muestran valores de MAP inferiores a los modelos RE+PRMS e IRE+PRMS. En la Figura 6.3a se los compara con respecto al modelo TF-IDF. De acuerdo a los índices de robustez, los modelos basados en MLM sufren de un mayor desvió de la consulta en comparación a aquellos que usan PRMS. Es decir, los términos de expansión de la consulta generados por los modelos RE+MLM e IRE+MLM tendría una menor relación a la consulta inicial, en comparación a los términos producidos por los modelos RE+PRMS e IRE+PRMS. Anteriormente se vio que los modelos de recuperación de entidades que utilizaban a MLM superaban el rendimiento de los aquellos que usaban PRMS. Una posible explicación a esto es la siguiente, los modelos de recuperación de documentos que utilizan a MLM generan una consulta expandida con términos poco relacionados a la consulta inicial, pero esto permitiría recuperar más entidades relevantes, ya que las entidades legales son representadas por documentos que contienen a otros documentos, es decir que se componen de muchos temas y por ello una consulta reformulada con términos sobre diversos temas favorecería la recuperación de entidades. Sin embargo, cuando se pretenden recuperar sumarios, los cuales tratan pocos temas, el uso del modelo MLM ya no resulta beneficioso.

[5]La curva de precisión y exhaustividad calcula el promedio de la *precisión promedio interpolada* en 11 niveles de exhaustividad $0.0, 0.1, \ldots, 1.0$ para cada necesidad de información en la colección de test [58, p. 146].



(a) TF-IDF, RE+MLM, IRE+MLM. (b) TF-IDF, RE+PRMS, IRE+PRMS.

Figura 6.3: Curvas de precisión y exhaustividad.

Colección	Métrica	RM3	RE	RE+MLM	RE+PRMS	IRE	IRE+MLM	IRE+PRMS
Sumarios20	MAP	0.313	0.310	0.299	0.335	0.318	0.312	**0.336**
	P@10	**0.280**	0.270	0.250	0.250	0.250	0.260	**0.280**
	P@20	0.200	0.195	0.180	0.210	**0.215**	0.205	0.209
	J@20	0.994	0.845	0.875	0.880	0.980	0.985	0.990
	RI	**0.10**	-0.10	-0.20	**0.10**	0	0	**0.10**

Tabla 6.12: Resultados de la evaluación de sistemas de recuperación de documentos con expansión de la consulta. El índice de robustez se calcula sobre TF-IDF.

En conclusión, los algoritmos de búsqueda con expansión de la consulta centrada en entidades que alcanzan el mayor rendimiento son los que utilizan al modelo PRMS. Los modelos RE+PRMS y IRE+PRMS tienen rendimientos similares, conocer cuáles son las entidades relevantes a una consulta, como lo hace el modelo IRE, parece no suponer una diferencia frente a la expansión automática de la consulta del modelo RE. Ambos superan el rendimiento del algoritmo TF-IDF.

Búsqueda en el SAIJ

Nos gustaría comparar el sistema de búsqueda del SAIJ con los algoritmos propuestos, pero esto no sería posible ya que la colección de documentos indexados en el SAIJ es distinta a la colección *Sumarios20* y el preprocesamiento de los datos utilizado en esta tesina difiere del realizado en el SAIJ. La colección con la cual venimos trabajando *Sumarios20* tiene cerca del 96 % de los sumarios de la base de datos del SAIJ y desconocemos de que manera se realizó la indexación en el SAIJ. Por lo tanto, la comparación no sería justa.

Sin embargo, podemos conocer cuál es el rendimiento del motor de búsqueda del SAIJ sobre la colección *Sumarios20* (o parte de ella) y comparar las distintas formas de búsqueda que ofrece el sistema. Entonces, se evalúan las búsquedas por palabras claves (o texto libre) conjuntivas y disyuntivas que provee el sistema del SAIJ, las cuales notamos como SAIJ_{and} y SAIJ_{or} respectivamente, sobre la colección *Sumarios20Nación*. Recordemos que las búsquedas en la colección *Sumarios20* pertenecen a la jurisdicción de Santa Fe o la Nación y como en SAIJ no es posible declarar una búsqueda con disyunción de jurisdicciones, se decide acotar las búsquedas solo a la jurisdicción de la Nación, ya que esta es la jurisdicción con más documentos relevantes. Por ello, se define la colección de test *Sumarios20Nación*, cuyos documentos son todos los sumarios del SAIJ sobre el

Colección	Métrica	$SAIJ_{and}$	$SAIJ_{or}$
Sumarios20Nación	MAP	0.020	0.055
	P@10	0.020	0.050
	J@10	0.880	0.880

Tabla 6.13: Recuperación de sumarios del sistema del búsqueda del SAIJ.

Derecho Civil, con jurisdicción en la Nación, hasta el año 2020, se trata de subconjunto de los documentos de *Sumarios20* y con los juicios de relevancia de *Sumarios20* referidos a la Nación, es decir las consultas solo tienen como jurisdicción a la Nación y no tiene documentos relevantes con jurisdicción de Santa Fe.

La pregunta sobre esta evaluación es la siguiente:

- ¿Cómo es el rendimiento de la búsqueda en el sistema del SAIJ?.

En la Tabla 6.13 se muestran los resultados de dicha evaluación. Se observa que los primeros diez resultados son poco precisos, aún cuando la mayoría de ellos fueron mostrados al jurado, ningún método de búsqueda supera el valor 0.1. Encontramos dos razones que explicarían estos resultados, primero es la existencia de un sesgo a favor de los algoritmos de búsquedas evaluados en secciones anteriores, el 24 % de los documentos que conforman el *pooling* provienen del sistema de búsqueda del SAIJ, y segundo, el SAIJ no indexó exclusivamente a la colección *Sumarios20Nación* sino probablemente un conjunto más grande de documentos.

Por lo menos, sabemos que los primeros 10 resultados de las búsquedas del SAIJ son poco precisas. Además, se observa que el uso del operador *or* alcanza un rendimiento superior al operador *and*.

Si bien la capacidad de generalizar las comparaciones de los algoritmos de búsquedas, presentados en este capítulo, para una colección arbitraria es limitada, a causa de los pocos juicios de relevancia utilizados, podemos brindar algunas conclusiones que se aplicarían para, por lo menos, la colección de test *Sumarios20*. Si se realiza una indexación de acuerdo a los tipos de documentos y los temas jurídico generales con los que fueron indexados semánticamente, como se hizo en esta tesina —aquí se indexó a los sumarios sobre el Derecho Civil, con jurisdicción en Santa Fe y la Nación—, se aplica el preprocesamiento de los datos detallados al principio de este capítulo y se utiliza un algoritmo de ranqueo tradicional sobre *Sumarios20*, se obtendrían resultados similares a los presentados en la Tabla 6.11. Además, si se utiliza un algoritmo de búsqueda con expansión de la consulta centrada en entidades, interactivo o automático, se mejoraría el rendimiento ante un algoritmo de búsqueda tradicional, las consultas beneficiadas por la expansión serian mayores a las perjudicadas. Los algoritmos propuestos alcanzarían rendimientos similares, sin embargo, suponemos que el algoritmo de expansión interactivo tendría la capacidad de guiar al usuario si este quisiera explorar los datos. Por último, encontramos que las búsquedas por palabras claves del sistema SAIJ son poco precisas, su rendimiento podría mejorar si utilizase como operador booleano al *or* —por defecto las palabras claves se expresan como conjunciones de términos—, y aplicase un mecanismo de indexación y preprocesamiento de los datos como el expuesto en esta tesina.

Capítulo 7

Conclusiones

En este trabajo, se propone la arquitectura de un sistema de búsqueda con expansión de la consulta para mejorar la recuperación de documentos. Este sistema permite incorporar como fuente de información a una base de conocimiento, de modo tal de expandir la consulta a partir de los términos asociados a las entidades relevantes de la consulta. Se implementaron dos algoritmos de búsqueda no supervisados basados en el Modelo de Lenguaje de Conceptos. Uno de ellos es el Modelo de Relevancia con Entidades, donde la selección de la entidades se realiza de manera automática. Mientras que el otro es el Modelo Iterativo de Relevancia con Entidades, el cual requiere la asistencia del usuario para seleccionar a las entidades que éste considere como relevantes. Una ventaja que proporciona la expansión interactiva frente a la automática es la capacidad de proveer sugerencias de búsquedas.

Los modelos propuestos se evaluaron en el dominio legal, sobre el ámbito del Derecho Civil, utilizando una base de conocimiento legal (*LegalBase*) y dos colecciones de test, (*Sumarios20* y *Entidades20*), desarrolladas especialmente para esta tesina. Se definió un tesauro de los accidentes de tránsito, reutilizando el tesauro del SAIJ, junto con la ontología *LegalOnto*, la cual modela la indexación semántica de los documentos legales. Estas fuentes de información más el tesauro del SAIJ fueron integradas para crear la base de conocimiento *LegalBase*. Esta fue poblada con los sumarios de la colección *Sumarios20*. Por lo tanto, *LegalBase* contiene sumarios del SAIJ, sus indexaciones semánticas y toda información de los dos tesauros legales, sus instancias, propiedades y clases. Cabe señalar que esta ontología legal puede utilizarse para otras aplicaciones.

En la evaluación, se encontró que el modelo RE supera al modelo IRE en la colección *Sumarios20*, posiblemente esto se deba a la falta de utilidad que tienen los conceptos que el usuario considera como relevantes, los términos asociados a tales entidades no son lo suficientemente buenos para la expansión de la consulta. También, se probaron algunas extensiones a los modelos propuestos, (1) una selección de entidades con mayor información semántica, donde se tiene en cuenta para el ranking de las entidades no sólo a la consulta basada en términos si no además, a las entidades relevantes de dicha consulta junto a sus tipos, y (2) distintas representaciones del modelo de lenguaje de las entidades que proveen de una mayor estructura a las descripciones de las entidades, nos referimos a los modelos MLM y PRMS. El uso del modelo PRMS, tanto en los algoritmos RE como IRE, alcanza un mayor rendimiento que el modelo MLM, aunque PRMS no tenga una cantidad de información uniforme en los campos de las descripciones de las entidades —esto sí puede decrementar su rendimiento cuando se lo utiliza en la recuperación de entidades—. Mientras que la selección semántica de entidades mejoraría al modelo RE, en el modelo IRE esto no sucede. La selección de entidades interactiva acota el beneficio que podría generar el uso de mejores entidades para la expansión de la consulta, si éstas

no son consideradas como relevantes por el usuario.

En la colección de sumarios *Sumarios20* el modelo RE muestra mejores resultados que su versión iterativa, IRE. Se debe tener en cuenta que los resultados mostrados están sujetos a un espacio de parámetros que no fue totalmente optimizado. Se lo restringió por limitaciones en el hardware, además la colección de test *Sumario20* es muy pequeña como para que resulte representativa, con sólo 10 juicios relevancia. Para demostrar la superioridad de un algoritmo frente al otro, es necesario realizar más pruebas sobre otras colecciones más grandes.

A través de este sistema de búsqueda semántico se espera ayudar a los profesionales del derecho en la recuperación de documentos relevantes para la elaboración de una demanda, por ejemplo. Como trabajo futuro se propone incluir modelos de lenguaje neuronales, como *word embeddings* [105], los cuales capturan relaciones semánticas entre las palabras, utilizar modelos de recuperación que integren a las entidades como una representación en paralelo de las representación basadas en términos [101] y extender el soporte a la búsqueda para otras ramas del derecho.

Bibliografía

[1] Proyecto ELI: European Legislation Identifier. Especificación Técnica para la Implementación del Identificador Europeo de Legislación en España (Fase 1).

[2] OWL 2 web ontology language document overview (second edition). W3C recommendation, W3C, December 2012. https://www.w3.org/TR/2012/REC-owl2-overview-20121211/.

[3] JeanAitchison, AlanGilchrist, and DavidBawden. *Thesaurus Construction and Use: a Practical Manual, 4th ed.*, 2002.

[4] Dean Allemang, Jim Hendler, and Fabien Gandon. *Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL*. Association for Computing Machinery, New York, NY, USA, 3 edition, 2020.

[5] Hiteshwar Kumar Azad and Akshay Deepak. Query expansion techniques for information retrieval: A survey. *Information Processing & Management*, 56(5):1698 – 1735, 2019.

[6] Franz Baader, Diego Calvanese, Deborah Mcguinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, Implementation, and Applications*. 01 2007.

[7] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Pearson Addison Wesley, Harlow, England, 2 edition, 2011.

[8] Thomas Baker, Sean Bechhofer, Antoine Isaac, Alistair Miles, Guus Schreiber, and Ed Summers. Key choices in the design of simple knowledge organization system (skos). *Journal of Web Semantics*, 20:35–49, 2013.

[9] Krisztian Balog. *Entity-Oriented Search*, volume 39 of *The Information Retrieval Series*. Springer, 2018.

[10] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, page 43–50, New York, NY, USA, 2006. Association for Computing Machinery.

[11] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. A language modeling framework for expert finding. *Information Processing & Management*, 45(1):1–19, 2009.

[12] Krisztian Balog, Marc Bron, and Maarten De Rijke. Query modeling for entity search based on terms, categories, and examples. *ACM Trans. Inf. Syst.*, 29(4), December 2011.

[13] Krisztian Balog, Yi Fang, Maarten de Rijke, Pavel Serdyukov, and Luo Si. Expertise retrieval. *Found. Trends Inf. Retr.*, 6(2–3):127–256, February 2012.

[14] Pedro M. et al. Bardi. *Tesauro SAIJ de Derecho Argentino.* Ediciones SAIJ, 2020.

[15] Keping Bi, Qingyao Ai, and W. Croft. *Iterative Relevance Feedback for Answer Passage Retrieval with Passage-Level Semantic Match*, pages 558–572. 04 2019.

[16] Keping Bi, Qingyao Ai, and W Bruce Croft. Revisiting iterative relevance feedback for document and passage retrieval. *arXiv preprint arXiv:1812.05731*, 2018.

[17] Roi Blanco, Peter Mika, and Sebastiano Vigna. Effective and efficient entity search in rdf data. pages 83–97, 01 2011.

[18] Agustín Aldo Capello et al. Sistema de recomendación para textos legales. B.S. thesis, 2018.

[19] Cristian Cardellino, Milagro Teruel, Rafael Gazzotti, Laura Alonso Alemany, and Serena Villata. Enhancing domain-specific ontologies at ease. In *III Simposio Argentino de Ontologías y sus Aplicaciones (SAOA)-JAIIO 46 (Córdoba, 2017).*, 2017.

[20] Fernando Cardellino, Cristian Cardellino, Karen Haag, Axel Soto, Milagro Teruel, Laura Alonso i Alemany, and Serena Villata. Mejora del acceso a infoleg mediante técnicas de procesamiento automático del lenguaje. In *XVIII Simposio Argentino de Informática y Derecho (SID)-JAIIO 47 (CABA, 2018)*, 2018.

[21] Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.*, 44(1), January 2012.

[22] Ilias Chalkidis, Charalampos Nikolaou, Panagiotis Soursos, and Manolis Koubarakis. Modeling and querying greek legislation using semantic web technologies. In *European Semantic Web Conference*, pages 591–606. Springer, 2017.

[23] Kuan-Yu Chen. Information retrieval and applications: Extended probabilistic models. `http://faculty.csie.ntust.edu.tw/~kychen/courses/2020_Fall_IR/2020-10-23%20Extended%20Probabilistic%20Models.pdf`, 2020.

[24] Stella G Dextre Clarke. The information retrieval thesaurus. *KO KNOWLEDGE ORGANIZATION*, 46(6):439–459, 2019.

[25] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley-Interscience, USA, 1991.

[26] W. B. Croft and D. J. Harper. Using probabilistic models of document retrieval without relevance information. *Journal of Documentation*, 35(4):285–295, January 1979. Publisher: MCB UP Ltd.

[27] W Bruce Croft, Donald Metzler, and Trevor Strohman. *Search engines: Information retrieval in practice*, volume 520. Addison-Wesley Reading, 2010.

[28] Gabriel A Dehner, Karina B Eckert, Juan M Lezcano, and Héctor J Ruidías. Modelo de recuperación de información jurídica basado en ontologías y distancias semánticas. In *XIX Simposio Argentino de Informática y Derecho (SID 2019)-JAIIO 48 (Salta)*, 2019.

[29] Stella Dextre Clarke and Marcia Zeng. From iso 2788 to iso 25964: the evolution of thesaurus standards towards interoperability and data modeling. *Information Standards Quarterly*, 24:20, 12 2013.

[30] Víctor Rodríguez Doncel and Elena Montiel Ponsoda. Lynx: Towards a legal knowledge graph for multilingual europe. *Law in Context. A Socio-legal Journal*, 37(1):1–4, 2020.

[31] Majirus Fansi. Fundamentals of Information Retrieval Illustration with Apache Lucene, 2012.

[32] Mariano Fernández-López, María Poveda-Villalón, Mari Carmen Suárez-Figueroa, and Asunción Gómez-Pérez. Why are ontologies not reused across the same domain? *Journal of Web Semantics*, 57:100492, 2019.

[33] Felipe N. Flores and Viviane P. Moreira. Assessing the impact of stemming accuracy on information retrieval – a multilingual perspective. *Information Processing & Management*, 52(5):840–854, 2016.

[34] Darío Garigliotti, Faegheh Hasibi, and Krisztian Balog. Identifying and exploiting target entity type information for ad hoc entity retrieval. *Information Retrieval Journal*, 22(3):285–323, 2019.

[35] Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho. *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer Science & Business Media, 2006.

[36] Audilio Gonzales-Aguilar, María Ramírez-Posada, and Diego Ferreyra. Tematres: software para gestionar tesauros. *Profesional de la Información*, 21(3):319–325, 2012.

[37] J. Graupmann, Jun Cai, and R. Schenkel. Automatic query refinement using mined semantic relations. In *International Workshop on Challenges in Web Information Retrieval and Integration*, pages 205–213, 2005.

[38] Stephan Grimm and Boris Motik. Closed world reasoning in the semantic web through epistemic operators. In *OWLED*, 2005.

[39] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. Exploiting entity linking in queries for entity retrieval. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ICTIR '16, page 209–218, New York, NY, USA, 2016. Association for Computing Machinery.

[40] Marti A. Hearst. Next generation web search: Setting our sites. *IEEE Data Engineering Bulletin*, 23, 2000.

[41] Marti A. Hearst. *Search User Interfaces*. Cambridge University Press, 2009.

[42] Djoerd Hiemstra and Wessel Kraaij. Twenty-one at trec-7: ad-hoc and cross-language track. In E.M Voorhees and D.K. Harman, editors, *Proceedings of the seventh Text Retrieval Conference (TREC)*, NIST Special Publications, pages 227–238, United States, 1999. National Institute of Standards and Technology.

[43] Aidan Hogan, Andreas Harth, and Axel Polleres. Saor: Authoritative reasoning for the web. In *Asian Semantic Web Conference*, pages 76–90. Springer, 2008.

[44] ISO 25964-1:2011: information, documentation Thesauri, and Thesauri for information retrieval interoperability with other vocabularies. Part 1. Standard, 2011.

[45] ISO 25964-2:2013: information, documentation Thesauri, and interoperability with other vocabularies. Part 2: Interoperability with other vocabularies. Standard, 2013.

[46] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.

[47] Chris Kamphuis, Arjen P. de Vries, Leonid Boytsov, and Jimmy Lin. Which bm25 do you mean? a large-scale reproducibility study of scoring variants. In Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins, editors, *Advances in Information Retrieval*, pages 28–34, Cham, 2020. Springer International Publishing.

[48] C Maria Keet. *An introduction to ontology engineering*. Maria Keet, 2018.

[49] Elisa F Kendall and Deborah L McGuinness. Ontology engineering. *Synthesis Lectures on The Semantic Web: Theory and Technology*, 9(1):i–102, 2019.

[50] Jinyoung Kim, Xiaobing Xue, and W. Bruce Croft. A probabilistic retrieval model for semistructured data. In Mohand Boughanem, Catherine Berrut, Josiane Mothe, and Chantal Soule-Dupuy, editors, *Advances in Information Retrieval*, pages 228–239, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[51] John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 111–119, New York, NY, USA, 2001. Association for Computing Machinery.

[52] Guiraude Lame. Using nlp techniques to identify legal ontology components: concepts and relations. In *Law and the Semantic Web*, pages 169–184. Springer, 2005.

[53] VICTOR LAVRENKO. *A GENERATIVE THEORY OF RELEVANCE*. PhD thesis, University of Massachusetts Amherst, 2004.

[54] Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 120–127, New York, NY, USA, 2001. Association for Computing Machinery.

[55] D. Leatherdale, Food, Agriculture Organization of the United Nations, and Commission of the European Communities. *Agrovoc: A Multilingual Thesaurus of Agricultural Terminology*. Number v. 1 in Agrovoc: A Multilingual Thesaurus of Agricultural Terminology. Apimondia, 1982.

[56] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. Pretrained transformers for text ranking: Bert and beyond. *arXiv preprint arXiv:2010.06467*, 2020.

[57] David E Losada, Javier Parapar, and Alvaro Barreiro. When to stop making relevance judgments? a study of stopping methods for building information retrieval test collections. *Journal of the Association for Information Science and Technology*, 70(1):49–60, 2019.

[58] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[59] M. Mercedes Martínez-González and María Alvite Díez. Thesauri and semantic web: Discussion of the evolution of thesauri toward their integration with the semantic web. *IEEE Access*, PP:1–1, 10 2019.

[60] Yosi Mass, Matan Mandelbrod, Einat Amitay, David Carmel, Yoelle Maarek, and Aya Soffer. Juruxml - an xml retrieval system at inex'02. pages 73–80, 01 2002.

[61] Fulvio Mazzocchi. Knowledge organization system (kos): an introductory critical account. *KO KNOWLEDGE ORGANIZATION*, 45(1):54–78, 2018.

[62] Deborah McGuinness and Frank van Harmelen. OWL web ontology language overview. W3C recommendation, W3C, February 2004. https://www.w3.org/TR/2004/REC-owl-features-20040210/.

[63] Edgar Meij, Dolf Trieschnigg, Maarten de Rijke, and Wessel Kraaij. Conceptual language models for domain-specific retrieval. *Information Processing & Management*, 46(4):448–469, 2010. Semantic Annotations in Information Retrieval.

[64] Massimo Melucci. *Boolean Model*, pages 260–260. Springer US, Boston, MA, 2009.

[65] Alistair Miles and Sean Bechhofer. SKOS simple knowledge organization system reference. W3C recommendation, W3C, August 2009. http://www.w3.org/TR/2009/REC-skos-reference-20090818/.

[66] David R. H. Miller, Tim Leek, and Richard M. Schwartz. A hidden markov model information retrieval system. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, page 214–221, New York, NY, USA, 1999. Association for Computing Machinery.

[67] Cristian Moral, Angélica de Antonio, Ricardo Imbert, and Jaime Ramírez. A survey of stemming algorithms in information retrieval. *Information Research: An International Electronic Journal*, 19(1):n1, 2014.

[68] Robert A N de Oliveira and Methanias C Junior. Experimental analysis of stemming on jurisprudential documents retrieval. *Information*, 9(2):28, 2018.

[69] Robert Neumayer, Krisztian Balog, and Kjetil Nørvåg. On the modeling of entities for ad-hoc entity search in the web of data. 01 2010.

[70] Natalya F Noy, Deborah L McGuinness, et al. Ontology development 101: A guide to creating your first ontology.

[71] Paul Ogilvie, , Paul Ogilvie, and Jamie Callan. Experiments using the lemur toolkit. In *In Proceedings of the Tenth Text Retrieval Conference (TREC-10*, pages 103–108, 2002.

[72] Arttu Oksanen, J Tuominen, E Mäkelä, M Tamper, Aki Hietanen, and Eero Hyvönen. Semantic finlex: Transforming, publishing, and using finnish legislation and case law as linked open data on the web. *Knowledge of the Law in the Big Data Age*, 317:212–228, 2019.

[73] Chris D. Paice. Method for evaluation of stemming algorithms based on error counting. *J. Am. Soc. Inf. Sci.*, 47(8):632–649, August 1996.

[74] Javier Parapar, David E. Losada, Manuel A. Presedo-Quindimil, and Alvaro Barreiro. Using score distributions to compare statistical significance tests for information retrieval evaluation. *Journal of the Association for Information Science and Technology*, 71(1):98–113, Apr 2019.

[75] Luciano Perezzini, Ana Casali, and Claudia Deco. Sistema de soporte para la recuperación de normativas en la ingeniería legal. In *XX Simposio Argentino de Informática y Derecho (SID 2020)-JAIIO 49 (Modalidad virtual)*, 2020.

[76] H. Sofia Pinto, Asuncion Gomez-Perez, and João Martins. Some issues on ontology integration. *Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving methods*, 06 1999.

[77] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 275–281, New York, NY, USA, 1998. Association for Computing Machinery.

[78] Martin F Porter. An algorithm for suffix stripping. *Program*, 1980.

[79] Eric Prud'hommeaux and Gavin Carothers. RDF 1.1 turtle. W3C recommendation, W3C, February 2014. https://www.w3.org/TR/2014/REC-turtle-20140225/.

[80] Ridho Reinanda, Edgar Meij, Maarten de Rijke, et al. Knowledge graphs: An information retrieval perspective. *Foundations and Trends in Information Retrieval*, 14(4), 2020.

[81] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, page 232–241, Berlin, Heidelberg, 1994. Springer-Verlag.

[82] Stephen Robertson. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation*, 60(5):503–520, January 2004. Publisher: Emerald Group Publishing Limited.

[83] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, April 2009.

[84] Stephen E Robertson. The probability ranking principle in ir. *Journal of documentation*, 1977.

[85] Stephen E Robertson and K Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146, 1976.

[86] J. J. Rocchio. *Relevance Feedback in Information Retrieval.* Prentice Hall, Englewood, Cliffs, New Jersey, 1971.

[87] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American society for information science*, 41(4):288–297, 1990.

[88] Gerard Salton, Edward A. Fox, and Harry Wu. Extended boolean information retrieval. *Commun. ACM*, 26(11):1022–1036, November 1983.

[89] Mark Sanderson and W Bruce Croft. The history of information retrieval research. *Proceedings of the IEEE*, 100(Special Centennial Issue):1444–1451, 2012.

[90] Christian Sifaqui. ¡poder a la gente! introduciendo servicios de open linked data a la base de datos legal de la biblioteca del congreso nacional de chile.

[91] Johan Smedt, Antoine Isaac, Stella Dextre Clarke, Jutta Lindenthal, Marcia Zeng, Douglas Tudhope, Leonard Will, and Vladimir Alexiev. Iso 25964 part 1: Thesauri for information retrieval: Rdf/owl vocabulary, extension of skos and skos-xl. Technical report, 2013.

[92] Ed Summers and Antoine Isaac. SKOS simple knowledge organization system primer. W3C note, W3C, August 2009. https://www.w3.org/TR/2009/NOTE-skos-primer-20090818/.

[93] Osma Suominen and Eero Hyvönen. Improving the quality of skos vocabularies with skosify. In *International Conference on Knowledge Engineering and Knowledge Management*, pages 383–397. Springer, 2012.

[94] Richard Fernando Tarupi Calan. Desarrollo de un sistema web para búsquedas de alto rendimiento sobre una base de datos de normas y estándares que se aplican en las áreas de ingeniería mediante la api lucene. B.S. thesis, 2019.

[95] Douglas Tudhope and Ceri Binding. Faceted thesauri. *Axiomathes*, 18(2):211–222, 2008.

[96] Jesús Vilares. Introducción a la recuperación de la información. `http://www.grupolys.org/docencia/ln/biblioteca/ir.pdf`, 2008.

[97] Gerhard Weikum, Luna Dong, Simon Razniewski, and Fabian Suchanek. Machine knowledge: Creation and curation of comprehensive knowledge bases, 2020.

[98] Leonard Will. The iso 25964 data model for the structure of an information retrieval thesaurus. *Bulletin of the American Society for Information Science and Technology*, 38(4):48–51, 2012.

[99] Chenyan Xiong and Jamie Callan. Esdrank: Connecting query and documents through external semi-structured data. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, page 951–960, New York, NY, USA, 2015. Association for Computing Machinery.

[100] Chenyan Xiong and Jamie Callan. Query expansion with freebase. In *Proceedings of the 2015 international conference on the theory of information retrieval*, pages 111–120, 2015.

[101] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. Bag-of-entities representation for ranking. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, pages 181–184, 2016.

[102] Helen Yannakoudakis. Information retrieval: Introduction and the boolean model. `https://www.cl.cam.ac.uk/teaching/1718/InfoRtrv/slides/lecture1-intro-boolean.pdf`, 2018.

[103] Liyang Yu. *A developer's guide to the semantic Web*. Springer Science & Business Media, 2011.

[104] Liyang Yu. *A Developer's Guide to the Semantic Web*. Springer, 2014.

[105] Hamed Zamani and W Bruce Croft. Embedding-based query language models. In *Proceedings of the 2016 ACM international conference on the theory of information retrieval*, pages 147–156, 2016.

[106] Hugo Zaragoza, Nick Craswell, Michael Taylor, Suchi Saria, and Stephen Robertson. Microsoft cambridge at trec-13: Web and hard tracks. In *IN PROCEEDINGS OF TREC 2004*, 2004.

[107] ChengXiang Zhai. Statistical language models for information retrieval a critical review. *Found. Trends Inf. Retr.*, 2(3):137–213, March 2008.

[108] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, page 403–410, New York, NY, USA, 2001. Association for Computing Machinery.

[109] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 334–342, New York, NY, USA, 2001. Association for Computing Machinery.

[110] Nikita Zhiltsov, Alexander Kotov, and Fedor Nikolaev. Fielded sequential dependence model for ad-hoc entity retrieval in the web of data. 08 2015.

[111] Xiaojin Zhu. CS838-1 Advanced NLP: Language Modeling. page 7.