UNIVERSIDAD DE SANTIAGO DE COMPOSTELA

Tecnologías de Gestión de Información No Estructurada

Luis Ardévol Mesa

Profesor:
Losada

Escola Técnica Superior de Enxeñaría
Master en Tecnoloxías de Análise
de Datos Masivos: Big Data

Curso 2024-2025

Contents

1 Introducción

El avance en la capacidad de generar y almacenar datos ha llevado a la necesidad de transformar los datos crudos en conocimiento accionable que pueda optimizar la toma de decisiones en diversos sectores como salud, seguridad, educación, ciencia y negocios (BI). Para lograrlo, es fundamental desarrollar tecnologías que permitan ver y extraer información útil y conocimiento oculto en los datos, particularmente en grandes cantidades de datos textuales. Gestionar y analizar grandes cantidades de datos textuales puede ayudar a los usuarios a gestionar y hacer uso de este tipo de datos en todo tipo de aplicaciones.

1.1 Datos textuales como *Big Data*

La gestión de texto de lenguaje natural, que abarca desde páginas web y redes sociales hasta literatura científica y documentos gubernamentales, se ha convertido en una prioridad debido a la explosión de datos en temas de todo tipo. Como un tipo especial de *big data*, los datos textuales representan una gran oportunidad para el descubrimiento de conocimiento y optimización de decisiones en múltiples aplicaciones. Un ejemplo de ello son los datos textuales de opiniones (valoraciones de productos, foros, redes sociales, ...).

Dado el volumen creciente de datos textuales, es imposible para una persona consumir toda la información relevante en tiempo real, de modo que se requieren sistemas inteligentes de recuperación de información para facilitar el acceso rápido a los datos necesarios. El mundo genera entre 1 y 2 *exabytes* de datos anualmente, de los cuales una gran cantidad es textual.

Atendiendo a la estructura de los datos, se pueden clasificar en dos tipos:

- Los datos estructurados, que cuentan con esquemas bien definidos, son manejados fácilmente por los ordenadores.
- Los datos no estructurados, como el texto, necesitan procesamiento computacional para interpretar su contenido, al tener una estructura menos explícita.

El procesamiento de lenguaje natural (NLP) aun no ha alcanzado un punto que permita al ordenador entender de forma precisa el texto. Generalmente se usan aproximaciones estadísticas y heurísticas para extraer información y analizar los datos textuales. El texto es de las fuentes de información más útiles ya que

- Es la forma más natural de codificar el conocimiento humano. Por ejemplo, el conocimiento científico casi solo existe en literatura científica, los manuales técnicos dan explicaciones detalladas de como operar aparatos, ...
- Es el tipo de información más frecuente.
- Es la forma de información más expresiva.

1.1.1 Recuperación y minería de texto

Para gestionar y explotar grandes cantidades de datos textuales, se suele recurrir a dos técnicas. Se aplica la recuperación de textos sobre una gran cantidad de datos textuales para, sobre los datos relevantes extraídos, aplicar minería y obtener conocimiento que usar en distintas aplicaciones.

Recuperación de texto (*text retrieval*)

No se puede digerir toda la cantidad de información disponible, por lo que son necesarios sistemas inteligentes de recuperación de información para facilitar el acceso rápido a los datos necesarios. Para esto surgen los motores de búsqueda, útiles en cualquier contexto donde haya una gran cantidad de datos textuales, no solo en la *web*.

Minería de texto (DM)

Los datos textuales son ricos en contenido semántico. La minería de texto busca descubrir conocimiento valioso dentro del contenido textual, usando herramientas de *software* inteligentes para descubrir patrones interesantes y opiniones que optimicen la toma de decisiones. El proceso de minería de datos puede describirse como minar los datos textuales para descubrir conocimiento útil.

La minería de datos aún no es tan madura como los motores de búsqueda, ya que el texto tiene una estructura menos explícita. El desarrollo de minería inteligente requiere que los ordenadores entiendan el contenido codificado en el texto.

1.1.2 Modos de acceso a la información: Pull vs Push

En el modo *pull*, el usuario toma la iniciativa de buscar información en el sistema, mientras que este último juega un papel pasivo y espera la petición del usuario. En las consultas, el usuario especifica la información necesaria y el sistema devuelve documentos que considera relevantes.

En el modo *push*, el sistema recomienda información anticipando las preferencias y necesidades de información del usuario. Suele funcionar bien cuando el usuario tiene una necesidad de información relativamente estable, como un *hobby*. Aquí, el sistema puede conocer las preferencias e intereses del usuario con adelanto.

En la navegación, el usuario se mueve por estructuras que enlazan elementos de información y alcanza información relevante progresivamente. La navegación y las consultas se alternan de forma natural.

1.2 Sistemas de Información Textual (TIS)

Estos sistemas buscan dar acceso a la información: conectan la información adecuada con el usuario adecuado en el momento adecuado. Ejemplos clásicos son:

- Motores de búsqueda: permiten al usuario acceder a información textual a través de consultas.
- Sistemas de recomendación: pueden sugerir información relevante al usuario de forma proactiva.

Se debe realizar un análisis de texto suficiente como para emparejar la información relevante con la información que necesita el usuario. Los elementos de información originales se muestran en su forma original, aunque en ocasiones se muestra a través de resúmenes dinámicos que dependen de la búsqueda del usuario (*snippets*).

1.2.1 Análisis del texto

Un análisis del texto permite adquirir el conocimiento útil codificado en los datos textuales, el cuál no es fácil de obtener sin sintetizar y analizar una gran cantidad de los datos. Por ejemplo, un motor de búsqueda simplemente devuelve las valoraciones relevantes de un producto, mientras que un motor de análisis extrae las opiniones positivas y negativas y compara opiniones de multitud de productos.

Los sistemas de información textual anotan una colección de documentos textuales con estructuras (tópicos) relevantes. Estas estructuras añadidas permiten al usuario buscar con restricciones sobre las mismas o navegar siguiéndolas.

A diferencia del DM, que se mueve bajo la premisa de descubrir y extraer patrones interesantes en los datos textuales, el NLP se mueve bajo la premisa de entender del texto de lenguaje natural de forma parcial, convertirlo en una forma de representación de conocimiento y hacer inferencia basándose en el conocimiento extraido.

1.2.2 Sistemas de información textual

Los TIS integran servicios de análisis de contenido basados en procesamiento de lenguaje natural (NLP) para transformar datos textuales crudos en representaciones más significativas, apoyando así la recuperación, categorización y organización de información. Se suele combinar el aprendizaje automático estadístico con conocimiento linguístico limitado. Las técnicas poco profundas son robustas, pero un análisis semántico profundo solo es factible en dominios específicos.

Algunas habilidades, como la de resumir documentos, requieren técnicas de NLP más profundas que otras, como una simple búsqueda. Sin embargo, la mayorías de TIS usan técnicas poco profundas, como bolsas de palabras.

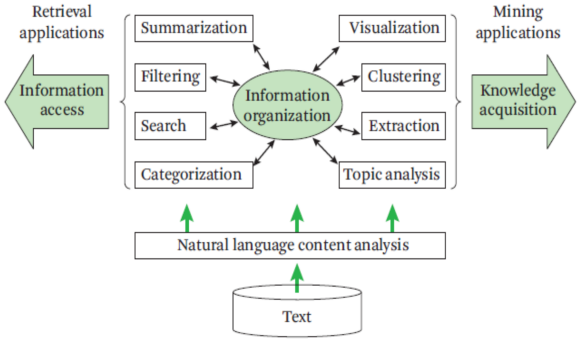

FIGURE I: Esquema conceptual de un TIS

Existen varias formas de organizar la información en un TIS:

- **Búsqueda**: toma una consulta del usuario y devuelve documentos relevantes.
- **Filtrado y Recomendación**: monitoriza el flujo de datos, selecciona los ítems relevantes para los intereses del usuario y luego recomienda los items relevantes (o filtra los no relevantes). Un sistema de recomendación tiene como objetivo recomendar información relevante al usuario, mientras que un sistema de filtrado tiene como objetivo filtrar la información no relevante, permitiendo que el usuario mantenga solo los *items* relevantes.

- **Categorización**: clasifica un objeto textual en una o varias categorías predefinidas. Puede anotar los objetos con todo tipo de categorías significativas, enriqueciendo la representación de los datos textuales. En este caso hay dos tipos de errores, los falsos positivos y los falsos negativos; hay que ver que métrica de rendimiento se le da al clasificador, dando más o menos *penalty* a un error concreto. Estos sistemas son tradicionales, y se usaban para, por ejemplo, clasificar correos electrónicos como *spam*.

- **Resumen**: toma uno o varios documentos de texto y genera un resumen conciso de los contenidos esenciales. Reduce el esfuerzo humano de digerir grandes cantidades de información textual.

- **Análisis de temáticas**: toma una serie de documentos y extrae y analiza los temas en ellos. Los temas aydan a digerir la información y permiten navegar por el texto de forma cómoda. Se puede combinar con datos no textuales, como tiempo, localización u otros metadatos. De este modo, es capaz de generar patrones de interés (tendencias temporales de temáticas, distribución espacio-temporal del tópicos, etc). Es tecnología no supervisada, por lo que los temas no están predefinidos, los da el propio algoritmo.

- **Extracción de información**: Extrae entidades y relaciones de entidades con otras áreas de conocimiento.

- **Clustering**: descubre grupos de objetos textuales similares (términos, oraciones, documentos, etc). Ayuda a los usuarios a explorar un espacio de la información, y es útil para descrubrir *outliers*.

- **Visualización**: permite representar visualmente patrones en los datos textuales.

2 Datos textuales

2.1 Procesamiento de lenguaje natural

El procesamiento de lenguaje natural (NLP) se ocupa de desarrollar técnicas computacionales para permitir que una computadora entienda el significado del texto en lenguaje natural. El NLP es una base fundamental de los sistemas de información textual (TIS) porque la efectividad de un TIS en ayudar a los usuarios a acceder y analizar datos textuales depende en gran medida de qué tan bien el sistema pueda entender el contenido de los datos textuales. Por lo tanto, el análisis de contenido es lógicamente el primer paso en el análisis y gestión de datos textuales.

Mientras que un humano puede entender instantáneamente una oración en su idioma nativo, resulta bastante complicado para un ordenador. En general, esto puede involucrar las siguientes tareas.

- Análisis léxico: determina las unidades significativas básicas en un idioma, como palabras. Determina el significado de cada palabra y delimita las fronteras de las palabras. Esto último puede ser más complicado en idiomas como el chino.
- Análisis sintáctico: determinar cómo se relacionan las palabras entre ellas dentro de una oración. Permite obtener más conocimiento que un análisis léxico, pero no tiene por qué tener idea del significado; no hay conocimiento.
- Análisis semántico: determina el significado de una oración. Esto típicamente implica el cálculo del significado de una oración completa o una unidad mayor basada en los significados de las palabras y su estructura sintáctica. Aquí es cuando se empieza a obtener conocimiento real.
- Análisis pragmático: determina el significado en contexto, por ejemplo, inferir los actos de habla del lenguaje. Una comprensión más profunda del lenguaje natural que el análisis semántico es, por lo tanto, entender aún más el propósito en la comunicación.
- Análisis del discurso: analiza segmentos extensos de texto, considerando varias oraciones, conexiones entre ellas y el contexto en el que se encuentran, considerando el resto de oraciones. Esto es lo que se busca que haga la tecnología (“te lo digo para que hagas algo al respecto”). Este tipo de análisis es muy complicado para textos de lenguaje natural no restringidos, y la tecnología “pre *deep learning*” lo hacía muy mal.

En la figura II, se muestra lo que implica entender una oración muy simple en inglés: *A dog is chansing a boy on the playground*. El análisis léxico en este caso implica determinar las categorías sintácticas (partes del discurso) de todas las palabras (por ejemplo, “*dog*” es un sustantivo y “*chasing*” es un verbo). El análisis sintáctico consiste en determinar que “*a*” y “*boy*” forman una frase nominal. Lo mismo ocurre con “*the*” y “*playground*”, y “*on the playground*” es una frase preposicional. El análisis semántico consiste en mapear las frases nominales a entidades y las frases verbales a relaciones para obtener una representación formal del significado de la oración. Por ejemplo, la frase nominal “*a boy*” puede mapearse a una entidad semántica que denota a un niño (es decir, b1), y “*a dog*” a una entidad que denota a un perro (es decir, d1). La frase verbal puede mapearse a un predicado de relación “*chasing*(d1, b1, p1)” como se muestra en la figura. Nótese que con este nivel de comprensión, también

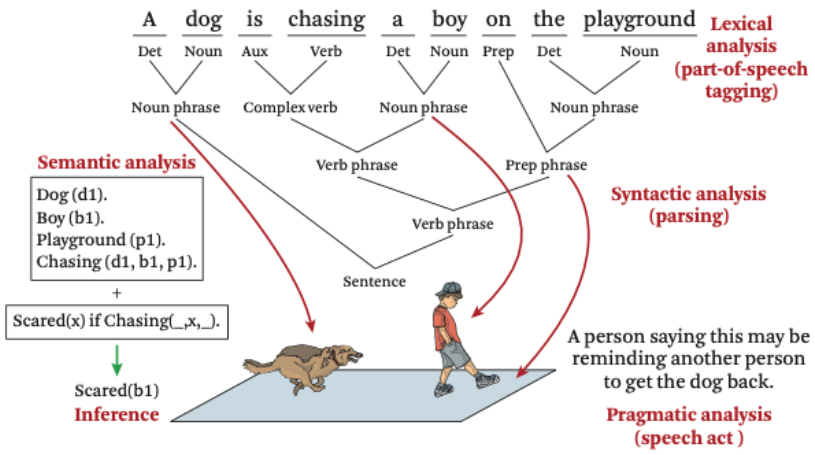

FIGURE II: Ejemplo de las tareas en el procesamiento de lenguaje natural.

se puede inferir información adicional basada en cualquier conocimiento de sentido común relevante. Por ejemplo, si se asume que si alguien está siendo perseguido puede estar asustado, se podría inferir que el niño que está siendo perseguido (b1) puede estar asustado. Finalmente, el análisis pragmático podría revelar además que la persona que dijo esta oración podría tener la intención de solicitar una acción, como recordar al dueño del perro que vigile al perro.

2.1.1 Desafíos y ambigüedades

Si bien es posible derivar una representación semántica clara para una oración simple como la que se muestra en la figura II, en general es muy cimplicado hacer este tipo de análisis para texto en lenguaje natural no restringido. La razón principal de esta dificultad es que el lenguaje natural está diseñado para hacer la comunicación humana eficiente; esto contrasta con un lenguaje de programación, que está diseñado para facilitar la comprensión por parte del ordenador. Específicamente, hay dos razones por las cuales el NLP es muy difícil.

- Se omite mucho conocimiento de sentido común en la comunicación en lenguaje natural porque se asumie que el receptor posee dicho conocimiento (por lo tanto, no hay necesidad de comunicarlo explícitamente).
- Se mantienen muchas ambigüedades, que se asumen que el receptor sabe cómo resolver (por lo tanto, no hay necesidad de gastar palabras para aclararlas). Como resultado, el texto en lenguaje natural está lleno de ambigüedad, y resolverla generalmente implicaría razonar con una gran cantidad de conocimiento de sentido común, lo cual es un desafío en inteligencia artificial.

En este sentido, el NLP es "completo en IA", es decir, tan difícil como cualquier otro problema complicado en inteligencia artificial.Algunos tipos de ambigüedades a las que hay que enfrentarse en NLP son:

- Ambigüedad a nivel de palabra: Una palabra puede tener múltiples categorías sintácticas o significados. Por ejemplo, “diseño” como sustantivo o verbo, (POS ambiguo) o palabras polisémicas (sentido ambiguo).
- Ambigüedad sintáctica: Las oraciones pueden tener múltiples estructuras sintácticas. Por ejemplo, procesamiento de lenguaje natural puede tener dos interpretaciones diferentes: “procesado del lenguaje natural” o “procesado natural del lenguaje” (modificación ambigua). “Un hombre vio a un niño con un telescopio,” que tiene interpretaciones diferentes que conducen a significados distintos (adjunción ambigua de frase preposicional (PP)).

- Resolución de anáforas: Determina a qué se refiere un pronombre, lo cual puede ser poco claro. "*John persuaded Bill to buy a TB for himself*", donde "himself" puede referirse a John o a Bill.
- Presuposición: Implica suposiciones, como en "Él ha dejado de fumar" implicando que fumaba antes. Hacer este tipo de inferencias resulta generalmente complicado.

2.1.2 Historia y estado del arte en NLP

La investigación en NLP se remonta al menos a la década de 1950, cuando los investigadores eran muy optimistas sobre tener ordenadores capaces de entender el lenguaje humano, particularmente con el propósito de la traducción automática. Sin embargo, pronto quedó claro que la traducción de alta calidad completamente automática no podría lograrse sin conocimiento. Un diccionario resultaba insuficiente; en su lugar, haría falta una enciclopedia.

Al darse cuenta de que la traducción automática podría ser demasiado ambiciosa, los investigadores abordaron aplicaciones menos ambiciosas de NLP a finales de la década de 1960 y 1970 con cierto éxito, aunque las técnicas desarrolladas no lograron escalar, teniendo así un impacto limitado en las aplicaciones. Por ejemplo, la gente probó aplicaciones de reconocimiento de voz donde el objetivo es transcribir un discurso. Esta tarea requiere solo una comprensión limitada del lenguaje natural, por lo tanto, es más realista. Se desarrollaron dos proyectos que demostraron la capacidad del ordenador para entender el lenguaje natural: uno es el proyecto Eliza, donde se utilizan reglas superficiales para permitir que un ordenador juegue el papel de un terapeuta para entablar un diálogo en lenguaje natural con un humano; el otro es el proyecto del mundo de bloques, que demostró la viabilidad de la comprensión semántica profunda del lenguaje natural cuando el lenguaje se limita a un dominio de juguete con solo bloques como objetos.

En las décadas de 1970 y 1980, se prestó atención al procesamiento de datos textuales en lenguaje natural del mundo real, particularmente a la comprensión de historias. Se desarrollaron muchos formalismos para la representación del conocimiento y reglas heurísticas de inferencia. Sin embargo, la conclusión general fue que incluso las historias simples son bastante difíciles de entender por un ordenador, confirmando la necesidad de una representación del conocimiento a gran escala e inferencias bajo incertidumbre.

Después de la década de 1980, los investigadores comenzaron a alejarse de los enfoques simbólicos tradicionales (basados en lógica) para el procesamiento del lenguaje natural, que en su mayoría habían demostrado no ser robustos para aplicaciones reales, y prestaron más atención a los enfoques estadísticos, que dieron más éxito; inicialmente en el reconocimiento de voz, y posteriormente en prácticamente el resto de tareas de NLP. A diferencia de los enfoques simbólicos, los enfoques estadísticos tienden a ser más robustos porque dependen menos de reglas generadas por humanos; en su lugar, a menudo aprovechan las regularidades y patrones en los usos empíricos del lenguaje, y se basan únicamente en datos de entrenamiento etiquetados por humanos y en la aplicación de técnicas de aprendizaje automático.

Si bien el conocimiento lingüístico siempre es útil, hoy en día, las técnicas de procesamiento del lenguaje natural más avanzadas tienden a depender en gran medida del uso intensivo de técnicas de aprendizaje automático estadístico, con el conocimiento lingüístico jugando solo un papel secundario. Estas técnicas de NLP estadístico son exitosas para algunas de las tareas de NLP.

2.1.3 Procesamiento de lenguaje natural estadístico

El NLP estadístico se basa en la probabilidad y en la estadística para resolver problemas de lenguaje natural. Algunas tareas comunes son:

- El etiquetado de partes del discurso (POS) es una tarea relativamente fácil, y los etiquetadores de POS en el estado del arte pueden tener una precisión muy alta (por encima del 97% en datos de noticias).
- El análisis sintáctico (*parsing*) es más difícil, aunque el análisis sintáctico parcial se puede hacer con una precisión razonablemente alta (por ejemplo, por encima del 90% para el reconocimiento de frases nominales).

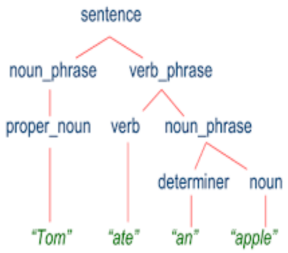

FIGURE III: Ejemplo de análisis sintáctico.

- Análisis completo de estructura (*full structure parsing*): es una tarea muy complicada debido a las ambigüedades en el lenguaje natural.
- Análisis semántico: asigna un significado a una oración. Es una tarea aún más complicada, con éxito limitado. Extracción de información notable(se reconocen entidades nombradas como nombres de personas y organizaciones, y relaciones entre entidades como quién trabaja en qué organización), la desambiguación del sentido de las palabras (distinguir diferentes sentidos de una palabra en diferentes contextos de uso) y el análisis de sentimientos (reconocer opiniones positivas sobre un producto en una reseña de producto) son áreas de interés. Inferencias y habla.
- Análisis de actos: determina la intención detrás de la comunicación. Generalmente, solo es factible en dominios muy limitados.

Procesamiento de lenguaje natural superficial y profundo

En textos arbitrarios, solo el análisis superficial del lenguaje natural se puede hacer de forma robusta. El análisis profundo no escala bien, y no es robusto para textos no restringidos. Este último, además, requiere una cantidad significativa de datos de entrenamiento (etiquetados por humanos) para obtener una precisión razonable.

2.2 Representación de texto

Las técnicas de NLP permiten diseñar muchos tipos diferentes de características informativas para los objetos textuales. Como ejemplo, la oración *A dog is chasing a boy on the playground* en la figura V. Se puede representar esta oración de muchas formas distintas. Primero, siempre se puede representar dicha oración como una cadena de caracteres. Esto es cierto para todos los idiomas, y es quizás la forma más general de representar texto, ya que siempre puede usarse este enfoque para representar cualquier dato textual. Desafortunadamente, la desventaja de esta representación es que no

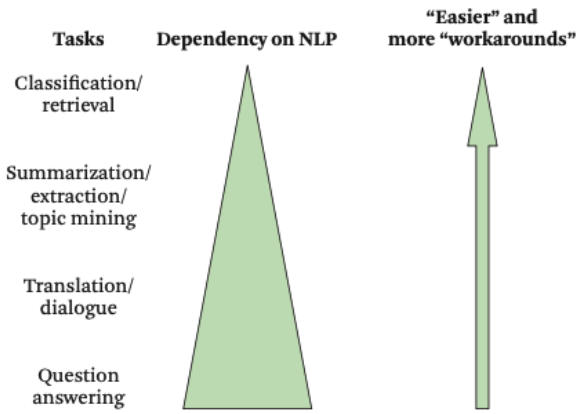

FIGURE IV: Dificultad de varias aplicaciones de NLP.

permite realizar análisis semántico, que a menudo es necesario en muchas aplicaciones de minería de texto. No se están reconociendo palabras, que son la unidad básica de significado para cualquier idioma.

La siguiente versión de la representación del texto es realizar la segmentación de palabras para obtener una secuencia de palabras. En la oración de ejemplo, se obtienen características como "*dog*" y "*chasing*". Con este nivel de representación se tiene mucha más libertad. Al identificar palabras, se puede, por ejemplo, descubrir fácilmente las palabras más frecuentes en un documento o en toda la colección. Estas palabras luego se pueden usar para formar temas. Por lo tanto, representar datos textuales como una secuencia de palabras abre muchas posibilidades de análisis interesantes.

Sin embargo, este nivel de representación es ligeramente menos general que una cadena de caracteres. En algunos idiomas, como el chino, no es tan fácil identificar todos los límites de las palabras. Para resolver este problema, se confia en algunas técnicas especiales para identificar palabras y realizar una segmentación más avanzada que no se base solo en los espacios en blanco (lo que no siempre es 100% preciso). Por lo tanto, la representación de la secuencia de palabras no es tan robusta como la representación de la cadena de caracteres. En inglés, es muy fácil obtener este nivel de representación, por lo que puede usarse todo el tiempo.

Avanzando más en el procesamiento del lenguaje natural, podemos agregar etiquetas de partes del discurso (POS) a las palabras. Esto permite contar, por ejemplo, los sustantivos más frecuentes, o determinar qué tipo de sustantivos están asociados con qué tipo de verbos. Esto abre más oportunidades para un análisis más profundo. Nótese en la figura V se usa un signo más en las características adicionales porque al representar el texto como una secuencia de etiquetas de partes del discurso, no necesariamente se reemplaza la secuencia de palabras original. En su lugar, se agrega esto como una forma adicional de representar datos textuales.

Representar el texto tanto como palabras como etiquetas de POS enriquece la representación de los datos textuales, permitiendo un análisis más profundo y fundamentado. Si se avanza más, entonces se estaría analizando la oración para obtener una estructura sintáctica. Esto abre más análisis de, por ejemplo, los estilos de escritura o la corrección de errores gramaticales.

Avanzando más en el análisis semántico, se podría reconocer "*dog*" como un animal. También podemos reconocer "*boy*" como una persona, y "*playground*" como una ubicación y analizar sus relaciones. Una deducción podría ser que el perro estaba persiguiendo al niño, y el niño está en el

parque. Esto agregará más entidades y relaciones, a través del reconocimiento de relaciones entre entidades. Ahora, se puede contar la persona más frecuente que aparece en toda esta colección de artículos de noticias. Estos tipos de patrones repetidos pueden potencialmente dar muy buenas características.

Esta representación de alto nivel es aún menos robusta que la secuencia de palabras o las etiquetas de POS, pero es muy útil. No siempre es fácil identificar todas las entidades con los tipos correctos y se pueden cometer errores. Las relaciones son aún más difíciles de encontrar. Si se mueve hacia una representación lógica, entonces existen predicados y reglas de inferencia. Con reglas de inferencia se pueden inferir hechos derivados interesantes del texto. No se puede hacer eso todo el tiempo para todo tipo de oraciones, ya que puede llevar un tiempo de computación significativo o una gran cantidad de datos de entrenamiento.

Finalmente, los actos de habla agregarían otro nivel de representación de la intención de esta oración. En este ejemplo, podría ser una solicitud. Saber eso permitiría analizar cosas aún más interesantes sobre el emisor de esta oración. ¿Cuál es la intención de decir eso? ¿Qué escenarios o qué tipos de acciones ocurrirán?

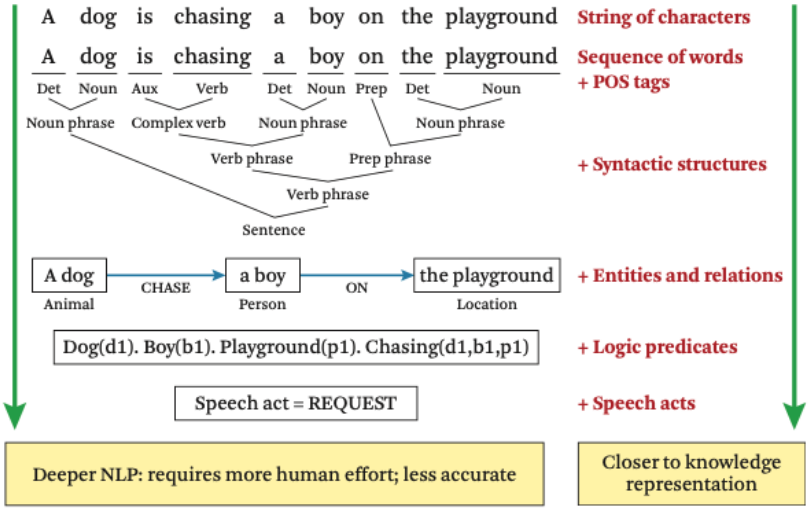

FIGURE V: Diferentes niveles de representación de texto.

Las técnicas de NLP más sofisticadas requieren más esfuerzo humano, y generalmente son menos robustas ya que intentan resolver un problema mucho más difícil. Si se analiza el texto en niveles que representan un análisis más profundo del lenguaje, entonces hay que tolerar posibles errores. Eso también significa que aún es necesario combinar dicho análisis profundo con análisis superficial basado en, por ejemplo, secuencias de palabras. A medida que se avanza, la representación del texto está más cerca de la representación del conocimiento en la mente humana; ese es el propósito de la minería de texto.

En la representación de textos hay que buscar un balance, un compromiso entre un análisis profundo, que puede dar errores pero dará un conocimiento directo que puede ser extraído del texto, y un análisis superficial, que es robusto pero no aportará una represención del conocimiento con el detalle adecuado.

Diferentes representaciones de texto tienden a permitir diferentes análisis, como se muestra en la figura VI. En particular, se agregan gradualmente resultados de análisis más profundos para representar datos textuales que abrirían más oportunidades de representación y capacidades de análisis.

Text Rep	Generality	Enabled Analysis	Examples of Application
String		String processing	Compression
Words		Word relation analysis; topic analysis; sentiment analysis	Thesaurus discovery; topic- and opinion-related applications
+ Syntactic structures		Syntactic graph analysis	Stylistic analysis; structure-based feature extraction
+ Entities & relations		Knowledge graph analysis; information network analysis	Discovery of knowledge and opinions about specific entities
+ Logic predicates		Integrative analysis of scattered knowledge; logic inference	Knowledge assistant for biologists

FIGURE VI: Representación de texto y análisis permitido.

2.3 Modelos de lenguaje estadísticos

Un modelo de lenguaje estadístico proporciona una distribución de probabilidad sobre secuencias de palabras, por ejemplo,

$$p(\textit{Today is Wednesday}) = 0.001$$
$$p(\textit{Today Wednesday is}) = 0.000000001$$
$$p(\textit{The equation has a solution}) = 0.000001$$

Un modelo de lenguaje puede depender del contexto. En el modelo de lenguaje mostrado anteriormente, la secuencia "*The equation has a solution*" tiene una probabilidad menor que "*Today is Wednesday*". Esto puede ser un modelo de lenguaje razonable para describir conversaciones generales, pero puede ser inexacto para describir conversaciones que ocurren en una conferencia de matemáticas, donde la secuencia "*The equation has a solution*" puede ocurrir con más frecuencia que "*Today is Wednesday*".

Sea un modelo de lenguaje, se pueden muestrear secuencias de palabras de acuerdo con la distribución para obtener una muestra de texto. En este sentido, podemos usar dicho modelo para "generar" texto. Por lo tanto, un modelo de lenguaje también se denomina a menudo un modelo generativo para texto.

2.3.1 Usos de modelos del lenguaje

- Reconocimiento de voz: Predice secuencias probables de palabras. Si se escucha, por ejemplo, *John feels*, se puede predecir que la siguiente palabra será *happy*, aunque haya palabras similares acústicamente, como *habit*. Esto es debido a que una es más probable que la otra.
- Categorización de texto: Determina la probabilidad de temas. Por ejemplo, si se tien *baseball* tres vces y *game* una en un artículo, ¿cómo de probable es que el tema sea "deportes"?

- Recuperación de información: Mejora la relevancia en búsquedas. Si un usuario está interesado en noticias de deportes, ¿cómo de probable es que se use la palabra *baseball* en una consulta?

2.3.2 Modelos de lenguaje

Si se enumeran todas las posibles secuencias de palabras y se asigna una probabilidad a cada secuencia, el modelo sería demasiado complejo de estimar, ya que el número de parámetros es potencialmente infinito al tener un número potencialmente infinito de secuencias de palabras. Es decir, nunca se dispondría de suficientes datos como para estimar estos parámetros. Por lo tanto, se deben hacer suposiciones para simplificar el modelo.

El modelo de lenguaje más simple es el modelo de lenguaje unigrama, en el cual se asume que una secuencia de palabras resulta de generar cada palabra de manera independiente. Por lo tanto, la probabilidad de una secuencia de palabras sería igual al producto de la probabilidad de cada palabra. Formalmente, sea V el conjunto de palabras en el vocabulario, y $w_1, \ldots, w_n$ una secuencia de palabras, donde $w_i \in V$ es una palabra. Así, la probabilidad de la secuencia de palabras sería

$$p(w_1, \ldots, w_n) = \prod_{i=1}^{n} p(w_i) \tag{2.1}$$

Dado un modelo de lenguaje unigrama θ, habrá tantos parámetros como palabras en el vocabulario, y estos satisfacen la restricción $\sum_{w \in V} p(w) = 1$. Tal modelo esencialmente especifica una distribución multinomial sobre todas las palabras.

Dado un modelo de lenguaje θ, en general, las probabilidades de generar dos documentos diferentes D_1 y D_2 serían diferentes, es decir, $p(D_1|\theta) \neq p(D_2|\theta)$. Intuitivamente, los documentos con mayor probabilidad serían aquellos que contienen muchas ocurrencias de las palabras de alta probabilidad según $p(w|\theta)$. En este sentido, las palabras de alta probabilidad de θ pueden indicar el tema capturado por θ.

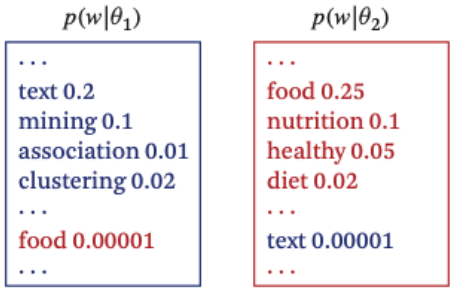

FIGURE VII: Dos ejemplos de modelos de lenguaje unigrama, representando dos temas distintos.

Por ejemplo, los dos modelos de lenguaje unigrama ilustrados en la Figura VII sugieren un tema sobre "minería de texto" y un tema sobre "salud", respectivamente. Intuitivamente, si D es un artículo sobre minería de texto, se esperaría que $p(D|\theta_1) > p(D|\theta_2)$, mientras que si D' es un artículo de blog que discute el control de la dieta, se esperaría lo contrario: $p(D'|\theta_1) < p(D'|\theta_2)$. También se puede esperar que $p(D|\theta_1) > p(D'|\theta_1)$ y $p(D|\theta_2) < p(D'|\theta_2)$.

Sea ahora un documento D que se asume que ha sido generado utilizando un modelo de lenguaje unigrama θ, y se quiere inferir el modelo subyacente θ (es decir, estimar las probabilidades de cada

palabra w, $p(w|\theta)$) basado en el documento observado D. Este es un problema estándar en estadística llamado estimación de parámetros y puede resolverse utilizando muchos métodos diferentes.

Un método popular es el estimador de máxima verosimilitud (ML), que busca un modelo $\hat{\theta}$ que daría a los datos observados la mayor verosimilitud (es decir, que mejor explique los datos):

$$\hat{\theta} = \arg\max_{\theta} p(D|\theta) \tag{2.2}$$

Es fácil demostrar que la estimación ML de un modelo de lenguaje unigrama da a cada palabra una probabilidad igual a su frecuencia relativa en D. Esto es,

$$p(w|\hat{\theta}) = \frac{c(w,D)}{|D|} \tag{2.3}$$

donde $c(w,D)$ es el conteo de la palabra w en D y $|D|$ es la longitud de D, o el número total de palabras en D.

Esta estimación es óptima en el sentido de que maximizaría la probabilidad de los datos observados, pero si realmente es adecuada para una aplicación sigue siendo cuestionable. Por ejemplo, si el objetivo es estimar el modelo de lenguaje en la mente de un autor de un artículo de investigación, y usamos el estimador de máxima verosimilitud para estimar el modelo basado solo en el resumen de un artículo, entonces claramente no es correcto, ya que el modelo estimado asignaría una probabilidad cero a cualquier palabra no vista en el resumen, lo que haría que todo el artículo tuviera una probabilidad cero a menos que solo use palabras del resumen. En general, la estimación de máxima verosimilitud asignaría una probabilidad cero a cualquier *token* o evento no observado en los datos; esto es así porque asignar una probabilidad no nula a dicho *token* quitaría masa de probabilidad que podría haberse asignado a una palabra observada (ya que todas las probabilidades deben sumar 1), reduciendo así la verosimilitud de los datos observados. Para mejorar el estimador de máxima verosimilitud se usan técnicas de suavizado.

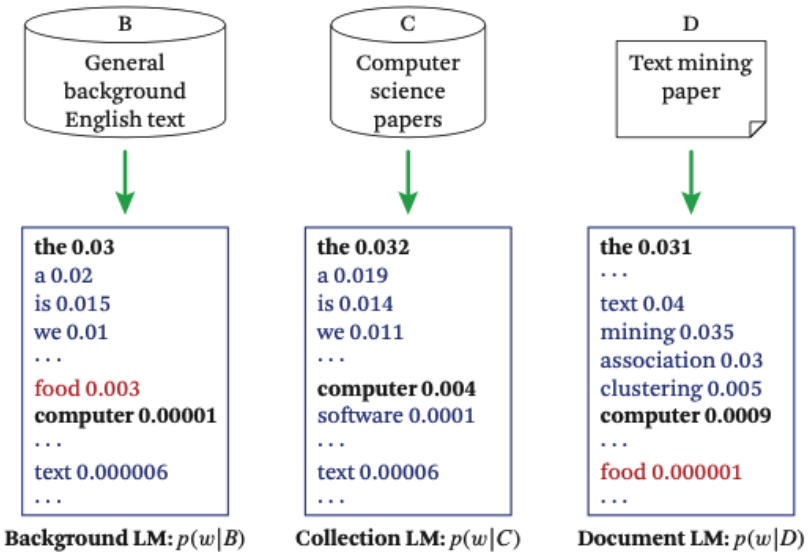

FIGURE VIII: Tres modelos de lenguaje diferentes representando tres temas distintos.

Aunque extremadamente simple, un modelo de lenguaje unigrama es muy útil para el análisis de texto. Por ejemplo, la figura VIII muestra tres modelos de lenguaje unigrama diferentes, estimados en

tres muestras distintas de datos textuales: una base de datos de texto en inglés general, una base de datos de artículos de investigación en ciencias de la computación y un artículo de investigación sobre minería de texto. En general, las palabras con las probabilidades más altas en los tres modelos son aquellas palabras funcionales en inglés, porque tales palabras se usan frecuentemente en cualquier texto. Bajando más en la lista de palabras, se verían más palabras con contenido y palabras temáticas. Las palabras de contenido serán completamente distintas dependiendo de los datos utilizados para la estimación y, por lo tanto, pueden usarse para discriminar los temas en diferentes muestras de texto.

Los modelos de lenguaje unigrama también pueden usarse para realizar análisis semántico de relaciones entre palabras. Por ejemplo, se pueden usarlos para encontrar qué palabras están asociadas semánticamente con una palabra como “computadora”. La idea principal para hacer esto es ver qué otras palabras tienden a co-ocurrir con esa palabra. Específicamente, primero se puede obtener una muestra de documentos (u oraciones) donde se menciona “computadora”. Luego se estima un modelo de lenguaje basado en esta muestra para obtener $p(w|\text{computadora})$. Este modelo dice qué palabras ocurren frecuentemente en el contexto de “computadora”. Sin embargo, las palabras más frecuentes según este modelo probablemente serían palabras funcionales en inglés o palabras que simplemente son comunes en los datos, sin una fuerte asociación con "computadora". Para filtrar las palabras comunes, se necesita un modelo para las mismas, que luego indique qué palabras deben ser filtradas.

Es fácil ver que el modelo de lenguaje inglés general (es decir, un modelo de lenguaje de fondo) serviría bien para el propósito. Se puede usar el modelo de lenguaje de fondo para normalizar el modelo $p(w|\text{computadora})$ y obtener una razón, un *ratio* de probabilidad para cada palabra. Las palabras con valores de razón altos pueden entonces asumirse como asociadas semánticamente con “computadora”, ya que tienden a ocurrir frecuentemente en su contexto, pero no en general. Esto se ilustra en la figura IX.

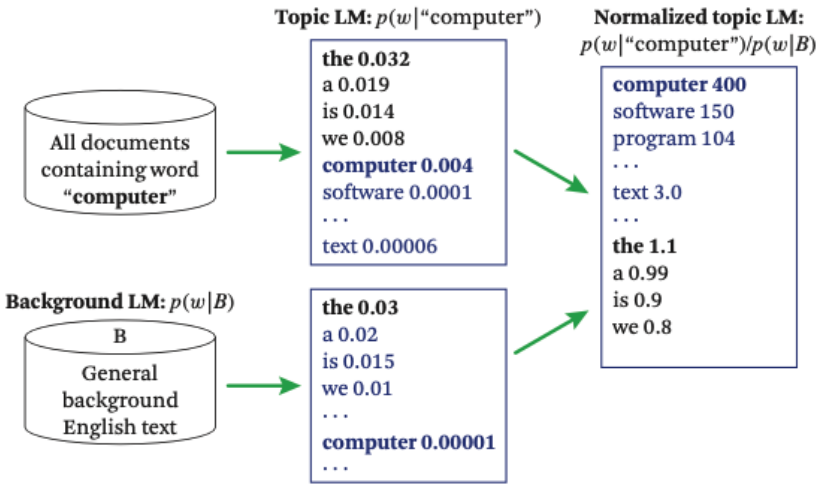

FIGURE IX: Uso de modelos de lenguaje temáticos y modelos de fondo para encontrar palabras semánticamente relacionada.

3 Visión general del acceso a datos textuales

El acceso a datos textuales es la base para el análisis de texto. La tecnología de acceso a texto desempeña dos roles importantes en las aplicaciones de gestión y análisis de texto. Primero, permite la recuperación de los datos textuales más relevantes para un problema de análisis particular, evitando así el procesamiento innecesario de una gran cantidad de datos no relevantes. Segundo, permite la interpretación de cualquier resultado de análisis o conocimiento descubierto en el contexto adecuado y proporciona la procedencia de los datos (origen).

El objetivo general del acceso a datos textuales es conectar a los usuarios con la información correcta en el momento adecuado. Esta conexión se puede realizar de dos maneras: *pull*, donde los usuarios toman la iniciativa de extraer información relevante del sistema, y *push*, donde el sistema toma la iniciativa de ofrecer información relevante a los usuarios.

3.1 Modos de acceso: *pull* y *push*

Dado que los datos textuales son creados para ser consumidos por humanos, estos últimos juegan un papel importante en las aplicaciones de análisis y gestión de datos textuales. Específicamente, los humanos pueden ayudar a seleccionar los datos más relevantes para un problema particular, lo cual es beneficioso al permitir evitar procesar la gran cantidad de datos textuales crudos (lo cual sería ineficiente) y centrarse en analizar la parte más relevante. Seleccionar datos textuales relevantes de una gran colección es la tarea básica del acceso a texto. Esta selección generalmente se basa en una especificación de la necesidad de información de un analista (un usuario), y se puede hacer en dos modos: pull y push. La figura X describe cómo estos modos se ajustan junto con la consulta y la navegación.

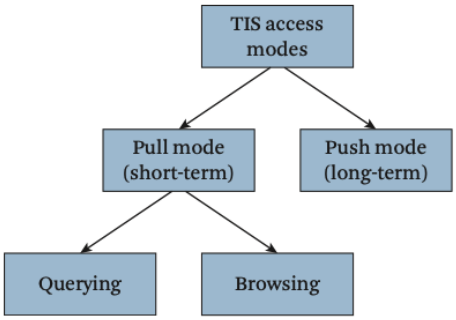

FIGURE X: La dicotomía de los modos de acceso a la información textual.

Pull

En el modo *pull*, el usuario inicia el proceso de acceso para encontrar los datos textuales relevantes, típicamente utilizando un motor de búsqueda. Este modo de acceso a texto es esencial cuando un usuario tiene una necesidad de información *ad hoc*, es decir, una necesidad de información temporal que podría desaparecer una vez que se satisfaga la necesidad. Por ejemplo, un usuario puede tener la necesidad de comprar un producto y, por lo tanto, estar interesado en recuperar todas las opiniones relevantes sobre productos candidatos; después de que el usuario haya comprado el producto, generalmente ya no necesitará dicha información. Otro ejemplo es que durante el proceso de análisis de datos de redes sociales para entender opiniones sobre un evento emergente, el analista también puede decidir explorar información sobre una entidad particular relacionada con el evento (por ejemplo, una persona), lo que también puede desencadenar una actividad de búsqueda.

Si bien la consulta es la forma más común de acceder a datos textuales en el modo *pull*, la navegación es otra forma complementaria de acceder a datos textuales en el modo *pull*, y puede ser muy útil cuando un usuario no sabe cómo formular una consulta efectiva, o encuentra inconveniente ingresar una consulta de palabras clave (por ejemplo, a través de un teléfono inteligente), o simplemente quiere explorar un tema sin un objetivo fijo. De hecho, al buscar en la *web*, los usuarios tienden a mezclar la consulta y la navegación (por ejemplo, al atravesar hipervínculos). En general, se puede considerar la consulta y la navegación como dos formas complementarias de encontrar información relevante en el espacio de información y, cuando la consulta no funciona bien, la navegación puede ser muy útil.

Push

En el modo *push*, el sistema inicia el proceso para recomendar un conjunto de elementos de información relevantes al usuario. Este modo de acceso a la información es generalmente más útil para satisfacer una necesidad de información a largo plazo de un usuario o analista, como pueden ser los intereses de investigación de un investigador, que pueden considerarse relativamente estables a lo largo del tiempo. En comparación, el flujo de información (es decir, los artículos de investigación publicados) es dinámico. Aunque un usuario puede buscar regularmente información relevante con consultas, es más deseable que un sistema de recomendación (también llamado sistema de filtrado) monitoree el flujo de información dinámico y "empuje" cualquier artículo relevante al usuario basado en la coincidencia de los artículos con los intereses del usuario (por ejemplo, en forma de un correo electrónico).

Otro escenario del modo *push* es la recomendación iniciada por el productor (difusión selectiva de información, SDI). En este escenario, el productor de información tiene interés en difundir la información entre los usuarios relevantes, y empujaría un elemento de información a dichos usuarios. La publicidad de información de productos en las páginas de resultados de búsqueda es un ejemplo de esto. La recomendación puede entregarse a través de notificaciones por correo electrónico o recomendarse a través de una página de resultados de un motor de búsqueda.

Necesidad de información

En términos generales, hay dos tipos de necesidades de información: a corto y a largo plazo. Las necesidades a corto plazo están a menudo asociadas con el modo *pull*, y las necesidades a largo plazo están más asociadas con el modo *push*. Una necesidad de información a corto plazo es temporal y generalmente se satisface a través de la búsqueda o navegación en el espacio de información, mientras que una necesidad de información a largo plazo puede satisfacerse mejor a través del filtrado o la

recomendación, donde el sistema tomaría la iniciativa de empujar la información relevante a un usuario.

La recuperación *ad hoc* es extremadamente importante porque las necesidades de información *ad hoc* aparecen con mucha más frecuencia que las necesidades de información a largo plazo. Las técnicas efectivas para la recuperación *ad hoc* generalmente pueden reutilizarse para el filtrado y la recomendación también. Además, en el caso de necesidades de información a largo plazo, es posible recopilar comentarios de los usuarios (*feedback*), que pueden ser explotados. En este sentido, la recuperación *ad hoc* es mucho más difícil, ya que no existe mucha información de retroalimentación de un usuario (es decir, pocos datos de entrenamiento para una consulta particular). Debido a la disponibilidad de datos de entrenamiento, el problema del filtrado o la recomendación generalmente puede resolverse utilizando técnicas de aprendizaje automático supervisado.

3.2 Acceso interactivo multimodal

Idealmente, el sistema debería proporcionar soporte para que los usuarios tengan acceso interactivo multimodal a datos textuales relevantes, de modo que los modos *push* y *pull* estén integrados en el mismo entorno de acceso a la información, y la consulta y la navegación también estén integradas sin problemas. Esto proporciona la máxima flexibilidad a los usuarios y les permitir consultar y navegar a voluntad.

FIGURE XI: Ejemplo de interfaz de navegación con un mapa de temas donde la navegación y la consulta están integradas de manera natural.

En la figura XI se tiene un sistema prototipo donde se ha añadido un mapa de temas construido automáticamente basado en un conjunto de consultas recopiladas en un motor de búsqueda comercial a una interfaz de motor de búsqueda regular para permitir que un usuario navegue por el espacio de información de manera flexible. Con esta interfaz, un usuario puede hacer cualquiera de las siguientes acciones en cualquier momento:

- Consulta (salto de largo alcance). Cuando un usuario envía una nueva consulta a través del cuadro de búsqueda, los resultados de un motor de búsqueda se muestran en el panel derecho. Al mismo tiempo, la parte relevante de un mapa de temas también se muestra en el panel izquierdo para facilitar la navegación si el usuario lo desea.

- Navegación en el mapa (caminata de corto alcance). El panel izquierdo de la interfaz permite que un usuario navegue por el mapa. Cuando un usuario hace clic en un nodo del mapa, este panel se actualizará y se mostrará una vista local con el nodo clicado como el foco actual. En la

vista local, mostramos los padres, los hijos y los vecinos horizontales del nodo actual en foco (etiquetado como "centro" en la interfaz). Un usuario puede así hacer *zoom* en un nodo hijo, hacer *zoom* hacia afuera a un nodo padre, o navegar a un nodo vecino horizontal. El número adjunto a un nodo es una puntuación para el nodo que usamos para clasificar los mismos. Dicho mapa permite al usuario "caminar" en el espacio de información para navegar por documentos relevantes sin necesidad de reformular consultas.

- Visualización de una región temática. El usuario puede hacer doble clic en un nodo temático en el mapa para ver los documentos cubiertos en la región temática. El panel de resultados de búsqueda se actualizaría con nuevos resultados correspondientes a los documentos en la región temática seleccionada.
- Visualización de un documento. Dentro del panel de resultados, un usuario puede seleccionar cualquier documento para verlo como en una interfaz de búsqueda estándar.

En la figura XII, se muestra un ejemplo de traza de navegación en la que el usuario comenzó con una consulta "*dining table*", hizo *zoom* en "*asian dining table*", hizo *zoom* hacia afuera de nuevo a "*dining table*", navegó horizontalmente primero a "*dining chair*" y luego a "*dining furniture*", y finalmente hizo *zoom* hacia afuera al tema general "*furniture*" donde el usuario tendría muchas opciones para explorar diferentes tipos de muebles. Si este usuario siente que se necesita un "salto largo", puede usar una nueva consulta para lograrlo. Dado que el mapa puede ocultarse y solo mostrarse cuando el usuario lo necesita, dicha interfaz es una extensión muy natural de la interfaz de búsqueda actual desde la perspectiva del usuario. Así, se puede ver cómo un sistema de acceso a texto puede combinar múltiples modos de acceso a la información para adaptarse a las necesidades actuales de un usuario.

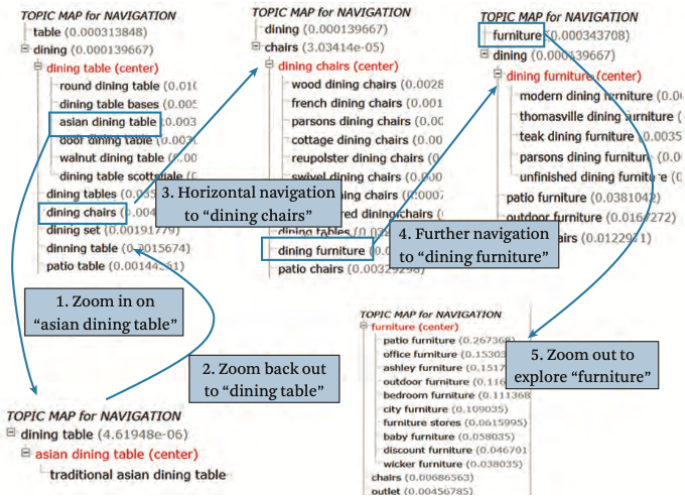

FIGURE XII: Ejemplo de traza de navegación en un mapa de temas, sin necesidad de hacer consultas.

3.3 Recuperación de texto

La herramienta más importante para apoyar el acceso a datos textuales es un motor de búsqueda (SE). Estos proporcionan soporte directo para la consulta y pueden extenderse fácilmente para proporcionar recomendaciones o navegación. Además, las técnicas utilizadas para implementar un motor de búsqueda efectivo a menudo también son útiles para la implementación de un sistema de recomendación, así como para muchas funciones de análisis de texto.

Desde la perspectiva del usuario, el problema de la TR es usar una consulta para encontrar documentos relevantes en una colección de documentos de texto. Esta es una tarea necesaria de forma frecuente, ya que los usuarios a menudo tienen necesidades de información *ad hoc* temporales para varias tareas, y necesitan encontrar la información relevante de inmediato. El sistema para apoyar la TR es un sistema de recuperación de texto, o un motor de búsqueda.

Aunque la TR a veces se usa indistintamente con el término más general "recuperación de información" (IR), este último también incluye la recuperación de otros tipos de información, como imágenes o videos. Sin embargo, las técnicas de recuperación para otros datos no textuales son menos maduras y, como resultado, la recuperación de esta tiende a depender del uso de técnicas de recuperación de texto para hacer coincidir una consulta de palabras clave con datos textuales acompañados de un elemento de datos no textuales. Por ejemplo, los motores de búsqueda de imágenes actuales en la *web* son esencialmente un sistema de TR donde cada imagen está representada por un documento de texto que consiste en cualquier dato textual asociado con la imagen (por ejemplo, título, leyenda, etc).

La tarea de la TR puede ser más o menos complicada dependiendo de consultas específicas y colecciones específicas. Por ejemplo, durante una búsqueda *web*, encontrar páginas de inicio generalmente es fácil, pero encontrar opiniones de personas sobre algún tema (por ejemplo, la política exterior de EE. UU.) sería más difícil. Hay varias razones por las cuales la TR es complicada:

- Una consulta suele ser bastante corta e incompleta (no es un lenguaje formal como SQL).
- La necesidad de información puede ser difícil de describir con precisión, especialmente cuando el usuario no está familiarizado con el tema.
- La comprensión precisa del contenido del documento es difícil. En general, dado que lo que cuenta como la respuesta correcta es subjetivo, incluso cuando los expertos humanos juzgan la relevancia de los documentos, pueden no estar de acuerdo entre sí.

Debido a la falta de estructuras semánticas claras y la dificultad en la comprensión del lenguaje natural, a menudo es un reto recuperar con precisión información relevante para la consulta de un usuario. De hecho, aunque los motores de búsqueda *web* actuales pueden parecer suficiente, aún puede ser complicado para un usuario localizar y recolectar rápidamente toda la información relevante para una tarea. En general, los motores de búsqueda actuales funcionan muy bien para consultas de navegación y consultas informativas simples y populares, pero en el caso de que un usuario tenga una necesidad de información compleja, como analizar opiniones sobre productos para comprar o investigar información médica sobre algunos síntomas, a menudo funcionan mal. Además, los motores de búsqueda actuales generalmente proporcionan poco o ningún soporte para ayudar a los usuarios a digerir y explotar la información recuperada. Como resultado, incluso si un motor de búsqueda puede recuperar la información más relevante, un usuario aún tendría que revisar una larga lista de documentos y leerlos en detalle para digerir completamente el conocimiento enterrado en los datos textuales para realizar su tarea.

3.4 Recuperación de texto vs recuperación de bases de datos

Es útil hacer una comparación del problema de la recuperación de texto (TR) y el problema de la recuperación de bases de datos. Ambas tareas de recuperación son para ayudar a los usuarios a encontrar información relevante, pero debido a la diferencia en los datos gestionados por estas dos tareas, hay muchas diferencias importantes.

Primero, los datos gestionados por un motor de búsqueda y un sistema de bases de datos son diferentes. En las bases de datos, los datos están estructurados, cada campo tiene un significado claramente definido según un esquema. Por lo tanto, los datos pueden verse como una tabla con columnas bien especificadas. Por ejemplo, en un sistema de base de datos de un banco, un campo puede ser nombres de clientes, otro puede ser la dirección, y otro más puede ser el saldo de cada tipo de cuenta. En contraste, los datos gestionados por un motor de búsqueda son texto no estructurado que puede ser difícil de entender para los ordenadores. Así, incluso si una oración dice que una persona vive en una dirección particular, sigue siendo difícil para el ordenador responder una consulta sobre la dirección de una persona en respuesta a una consulta de palabras clave, ya que no hay una estructura definida simple para el texto libre. Aquí es el sistema el que debe ver qué es relevante en cada búsqueda.

En segundo lugar, una consecuencia de la diferencia en los datos es que las consultas que pueden ser soportadas por los dos también son diferentes. Una consulta de base de datos especifica claramente las restricciones en los campos de la tabla de datos y, por lo tanto, los resultados de recuperación esperados (respuestas a la consulta) están muy bien especificados sin ambigüedad. En un motor de búsqueda, sin embargo, las consultas son generalmente palabras clave, que son solo una especificación vaga de qué documentos deben ser devueltos. Incluso si el ordenador puede entender completamente la semántica del texto en lenguaje natural, a menudo resulta que la necesidad de información del usuario es vaga debido a la falta de conocimiento completo sobre la información que se debe encontrar (lo cual es a menudo la razón por la que el usuario quiere encontrar la información en primer lugar). Por ejemplo, en el caso de buscar literatura relevante para un problema de investigación, es poco probable que el usuario pueda especificar clara y completamente qué documentos deben ser devueltos.

Finalmente, los resultados esperados en las dos aplicaciones también son diferentes. En la búsqueda en bases de datos, podemos recuperar elementos de datos muy específicos (por ejemplo, columnas específicas); en la TR, generalmente solo podemos recuperar un conjunto de documentos relevantes. Con pasajes o campos identificados en un documento de texto, un motor de búsqueda también puede recuperar pasajes, pero generalmente es difícil recuperar entidades específicas o valores de atributos como podemos en una base de datos. Esta diferencia no es tan esencial como la diferencia en la especificación vaga de cuál es exactamente la "respuesta correcta" a una consulta, pero es una consecuencia directa de la necesidad de información vaga en la TR.

Debido a estas diferencias, los desafíos en la construcción de una base de datos útil y un motor de búsqueda útil también son algo diferentes. En las bases de datos, dado que los elementos que deben ser devueltos están claramente especificados, no hay desafío en determinar qué elementos de datos satisfacen la consulta del usuario y, por lo tanto, deben ser devueltos; un desafío importante es cómo encontrar las respuestas lo más rápido posible, especialmente cuando hay muchas consultas siendo emitidas al mismo tiempo. Aunque el desafío de la eficiencia también existe en un motor de búsqueda, un desafío más importante es averiguar qué documentos deben ser devueltos para una consulta antes de preocuparse por cómo devolver las respuestas rápidamente. En las aplicaciones de bases de datos, también es muy importante mantener la integridad de los datos, es decir, asegurar que no ocurra ninguna inconsistencia debido a una falla de energía. En la TR, modelar la necesidad de información del usuario y las tareas de búsqueda es importante, nuevamente debido a la dificultad para un usuario de especificar claramente las necesidades de información y la dificultad en el procesamiento del lenguaje natural.

Dado que lo que cuenta como la mejor respuesta a una consulta depende del usuario, en la TR, el usuario es en realidad parte de nuestra entrada (junto con la consulta y el conjunto de documentos). Por lo tanto, no hay una manera matemática de probar que una respuesta es mejor que otra o probar

que un método es mejor que otro. En cambio, siempre tenemos que confiar en la evaluación empírica utilizando algunas colecciones de prueba y usuarios. En contraste, en la investigación de bases de datos, dado que el problema principal es la eficiencia, uno puede probar que un algoritmo es mejor que otro analizando la complejidad computacional o realizando algún estudio de simulación. Sin embargo, al realizar un estudio de simulación (para determinar qué algoritmo es más rápido), también se enfrenta el mismo problema que en la recuperación de texto: la simulación puede no reflejar con precisión las aplicaciones reales. Por lo tanto, un algoritmo que se demuestre más rápido con la simulación puede no ser realmente más rápido para una aplicación particular. De manera similar, un algoritmo de recuperación que se demuestre más efectivo con una colección de prueba puede resultar ser menos efectivo para una aplicación particular o incluso otra colección de prueba. Cómo evaluar de manera confiable los algoritmos de recuperación es en sí mismo un tema de investigación desafiante.

Debido a la diferencia, los dos campos han sido tradicionalmente estudiados en diferentes comunidades con una base de aplicación diferente. Las bases de datos han tenido aplicaciones generalizadas en prácticamente todos los dominios con una industria bien establecida y fuerte. La comunidad de IR que estudia la recuperación de texto ha sido una comunidad interdisciplinaria que involucra la ciencia de la información y la informática, pero no había tenido una base industrial fuerte hasta que nació la *web* a principios de la década de 1990. Desde entonces, la industria de los motores de búsqueda ha dominado, y a medida que más y más información en línea está disponible, las tecnologías de motores de búsqueda (que incluyen TR y otros componentes técnicos como el aprendizaje automático y el procesamiento del lenguaje natural) continuarán creciendo.

3.5 Selección y clasificación de documentos

Sea un colección de documentos (un conjunto de documentos de texto desordenados), la tarea de recuperación de texto (TR) puede definirse como el uso de una consulta de usuario (es decir, una descripción de la necesidad de información del usuario) para identificar un subconjunto de documentos que puedan satisfacer la necesidad de información del usuario.

Formalmente, sea $V = \{w_1, ..., w_N\}$ un conjunto de vocabulario de todas las palabras en un idioma natural particular donde w_i es una palabra. La consulta de un usuario $q = q_1, q_2, ..., q_m$ es una secuencia de palabras, donde $q_i \in V$. De manera similar, un documento $d_i = d_{i1}, ..., d_{im}$ también es una secuencia de palabras donde $d_{ij} \in V$. En general, una consulta es mucho más corta que un documento, ya que la consulta a menudo es escrita por un usuario utilizando un sistema de motor de búsqueda, y los usuarios generalmente no quieren hacer mucho esfuerzo para escribir muchas palabras. Sin embargo, esto no siempre es el caso (por ejemplo, en una búsqueda en *Twitter*, cada documento es un tweet.

La colección de textos $C = \{d_1, ..., d_M\}$ es un conjunto de documentos de texto. En general, se puede asumir que existe un subconjunto de documentos en la colección, es decir, $R(q) \subset C$, que son relevantes para la consulta del usuario q; es decir, son documentos relevantes o documentos útiles para el usuario que escribió la consulta. Naturalmente, este conjunto relevante depende de la consulta q. Sin embargo, qué documentos son relevantes generalmente es desconocido; la consulta del usuario es solo una "pista" de qué documentos deberían estar en el conjunto $R(q)$. Además, diferentes usuarios pueden usar la misma consulta para intentar recuperar conjuntos de documentos relevantes algo diferentes. Esto significa que es irreal esperar que un ordenador devuelva exactamente el conjunto $R(q)$, a diferencia del caso en la búsqueda en bases de datos, donde esto es factible. Por lo tanto, lo mejor que un ordenador puede hacer es devolver una aproximación de $R(q)$, $R'(q)$.

A un alto nivel, hay dos estrategias alternativas para obtener $R'(q)$: selección de documentos vs. clasificación de documentos.

En la selección de documentos, se implementa un clasificador binario para clasificar un documento como relevante o no relevante con respecto a una consulta particular. Es decir, se diseña una función de clasificación binaria, o una función indicadora, $f(q,d) \in \{0,1\}$. Si $f(q,d) = 1$, se asumirá que d es relevante, mientras que si $f(q,d) = 0$, no será relevante. Por lo tanto, $R'(q) = \{d | f(q,d) = 1, d \in C\}$. Usando esta estrategia, el sistema debe estimar la relevancia absoluta, es decir, si un documento es relevante o no.

Una estrategia alternativa es clasificar los documentos y dejar que el usuario decida un punto de corte. Es decir, se implementa una función de clasificación $f(q,d) \in \mathbb{R}$ y se calsifican todos los documentos en valores descendentes de esta función de clasificación. Un usuario navegará por la lista clasificada y se detendrá cuando lo considere apropiado. En este caso, el conjunto $R'(q)$ es en realidad definido en parte por el sistema y en parte por el usuario, ya que el usuario elegiría implícitamente un umbral de puntuación θ basado en la posición de clasificación donde se detuvo. En este caso, $R'(q) = \{d | f(q,d) \geq \theta\}$. Usando esta estrategia, el sistema solo necesita estimar la relevancia relativa de los documentos: qué documentos son más probablemente relevantes.

Dado que la estimación de la relevancia relativa es intuitivamente más fácil que la de la relevancia absoluta, se espera que sea más fácil implementar la estrategia de clasificación. De hecho, la clasificación generalmente se prefiere a la selección de documentos por múltiples razones:

- Debido a la dificultad para un usuario de prescribir los criterios exactos para seleccionar documentos relevantes, es poco probable que el clasificador binario sea preciso. A menudo, la consulta está sobre-restringida o sub-restringida.
  - En el caso de una consulta sobre-restringida, puede que no haya documentos relevantes que coincidan con todas las palabras de la consulta, por lo que forzar una decisión binaria puede resultar en no entregar ningún resultado de búsqueda.
  - Si la consulta está sub-restringida (demasiado general), puede haber demasiados documentos que coincidan con la consulta, lo que resulta en una sobre-entrega.

A menudo es muy difícil para un usuario conocer el nivel "correcto" de especificidad de antemano antes de explorar la colección de documentos. Incluso si el clasificador puede ser preciso, un usuario aún se beneficiaría de la priorización de los documentos relevantes coincidentes para el examen, ya que un usuario solo puede examinar un documento a la vez y algunos documentos relevantes pueden ser más útiles que otros (grados de relevancia). Por todas estas razones, clasificar documentos apropiadamente se convierte en un desafío técnico principal en el diseño de un sistema de recuperación de texto efectivo.

La estrategia de clasificación se muestra además como óptima teóricamente bajo dos suposiciones basadas en el principio de clasificación por probabilidad (Robertson, 1997), que establece que devolver una lista clasificada de documentos en orden descendente de relevancia predicha es la estrategia óptima bajo las siguientes dos suposiciones:

- La utilidad de un documento para un usuario es independiente de la utilidad de cualquier otro documento.
- Un usuario navegará por los resultados secuencialmente.

Entonces, el problema es el siguiente: se tiene una consulta que es una secuencia de palabras, y un documento que también es una secuencia de palabras, y se quiere definir la función $f(q,d)$, capaz

de calcular una puntuación basada en la consulta y el documento. El desafío principal es diseñar una buena función de clasificación que pueda clasificar todos los documentos relevantes por encima de los no relevantes.

Ahora, esto significa que la función debe ser capaz de medir la probabilidad de que un documento d sea relevante para una consulta q. Eso también significa que debe haber alguna forma de definir la relevancia.