Trabajo de Fin de Grado

Extracción y visualización de información de textos legales

Extraction and visualization of information from legal documents

Francisco Javier Rodríguez Dioniz

La Laguna, 7 de julio de2015

D. **Isabel Sánchez Berriel**, con N.I.F. 42.885.838-Sprofesora Colaboradora adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora.

D. **Marcos Colebrook Santamaría**, con N.I.F. 43.787.808-V profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor.

C E R T I F I C A (N)

Que la presente memoria titulada:

"Extracción y visualización de información de textos legales"

ha sido realizada bajo su dirección por D. **Francisco Javier Rodríguez Dioniz**, con N.I.F. 78.858.026-L.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 08 de julio de 2015.

Agradecimientos

A Isabel Sánchez Berriel y a Marcos Colebrook Santamaría por dedicar su tiempo y su apoyo a este proyecto y por todo lo que he aprendido de ellos en este trabajo.

A todos mis amigos y compañeros de grado, los cuales siempre me han animado y apoyado.

A mi familia por su apoyo y su esfuerzo. Gracias a ellos ha sido posible que estudie lo que me apasiona.

Licencia

Resumen

El objetivo de este trabajo ha sido la obtención de información estructurada a partir de corpus de textos de jurisprudencia para la posterior visualización de la misma, facilitando la obtención de conclusiones sobre las sentencias judiciales a partir de dicha información.

Los textos sobre los que se trabajan son obtenidos del Centro de Documentación Judicial (CENDOJ), órgano técnico del Consejo General del Poder Judicial.

Para ello, ha sido necesaria la extracción del texto de las sentencias judiciales y la utilización de técnicas de procesamiento del lenguaje natural. Las sentencias judiciales son clasificadas utilizando técnicas de similitud de documentos, haciendo posible categorizarlas como favorables, desfavorables o parciales.

Palabras clave: Legal, extracción de información, visualización, procesamiento del lenguaje natural.

Abstract

The goal of this project is to obtain structured information from the corpus of legal texts, for its subsequent visualization, and this, obtaining some conclusions about the court judgments.

The legal texts used for this project have been obtained from the Centro de Documentación Judicial (CENDOJ), which belongs to the Consejo General del Poder Judicial.

The development involved the extraction of texts from court judgments, as well as the use of natural language processing (NLP) techniques. Accordingly, the court judgments are classified using document similarity techniques in favorable, unfavorable and partially favorable.

Keywords: Legal, extraction of information, visualization, natural language processing (NLP).

Índice General

Índicedefiguras

Índice de tablas

Capítulo 1. Introducción

Este capítulo recoge la descripción del problema, definiendo los objetivos, el alcance y antecedentes del mismo.

1.1 Objetivo

El objetivo de este proyecto es desarrollar una herramienta que permita extraer de forma automática la información de sentencias judiciales y el tratamiento de dicha información para su posterior visualización.

En definitiva, la herramienta desarrollada extraerá y procesará la información necesaria de los textos jurídicos, la cual será utilizada para visualizar datos de las sentencias, juzgados y figuras judiciales.

Por lo tanto, se deben diferenciar dos puntos clave:

- Información básica
  - Extracción de figuras judiciales (Ponente, Magistrados, letrados, procuradores).
  - Extracción de los juzgados de origen.
- Parte resolutiva

  El fallo o parte resolutiva de la sentencia es la que contiene la decisión, condena o absolución del acusado.
  - Creación de una base de conocimiento clasificada en sentencias favorables, desfavorables y parciales.
  - Creación de glosario de frecuencias de palabras y de documento representante para cada clase.
  - Clasificador de nuevas sentencias, por similitud de documento entre la nueva sentencia y el representante de cada clase.

Finalmente, obtendremos dichos datos de forma que puedan ser preparados y tratados para ser visualizados.

1.2 Alcance

La herramienta desarrollada pretende realizar las siguientes tareas:

- Los textos sobre los que se trabajan son obtenidos del CENDOJ, acotando el ámbito de estudio en el siguiente:
  - Trabajamos sobre la jurisdicción Civil.
  - Sentencias judiciales del órgano Audiencia Provincial.
  - Provincias: Santa Cruz de Tenerife, Las Palmas de Gran Canaria y Madrid, si bien la solución es extensible a cualquier provincia española.
  - Tema que se expone: Divorcio.
- Extracción de la información de las sentencias.
- Creación de una base de conocimiento a partir de sentencias previamente clasificadas.
- Clasificación de las sentencias (favorables, desfavorables o parciales).
- Visualización de la cantidad de las sentencias clasificadas, siendo posible filtrarlas por las sedes en las que trabajamos (Santa cruz de Tenerife, Las Palmas de Gran Canaria o Madrid) y por los ponentes de cada una de dichas sedes.
- Visualización de la cantidad de sentencias de los juzgados de origen, las cuales han sido clasificadas como favorables y parciales en la Audiencia Provincial.
- La aplicación será capaz de trabajar con las sentencias que se han utilizado en dicho trabajo.

1.3 Destinatarios

El proyecto va dirigido especialmente a todo tipo de profesionales y miembros de la Carrera Judicial, así como aquellos que necesiten la utilización de este tipo de herramientas para uso profesional o uso propio.

Organizaciones que tengan relación con la Carrera Judicial o que tengan interés en el uso de los datos que se proporcionan.

También va dirigido a cualquiera que tenga especial interés en visualizar datos de las sentencias españolas y sacar conclusiones sobre ello.

1.4 Antecedentes

En el ámbito de este proyecto, se pueden encontrar diferentes aplicaciones desarrolladas. Las que más afinidades tienen con la que presentamos en la solución desarrollada:

RavelLaw(ravellaw.com) es una herramienta que permite la visualización de la información de casos importantes en EEUU, hace posible la búsqueda de patrones entre dichos casos, o entre jueces. Permite hacer consultas por palabras, devuelve el resultado de los casos donde salga dicha palabra, y también la relación entre ellos, mostrándolo de forma gráfica y en orden cronológico. Los objetivos de este trabajo de fin de grado no son exactamente iguales a los de RavelLaw, pero es una aplicación con la que se comparten características y que trata temas de sentencias de casos reales.

Lex Machina (lexmachina.com) extrae datos de casos judiciales, mostrando información sobre jueces, abogados, patentes, etc, extraídos de millones de páginas de información sobre casos de propiedad intelectual.

Counselytics (counselytics.com) obtiene análisis de documentos identificando la presencia o ausencia de conceptos legales importantes y organiza los documentos analizados para un acceso o gestión futuro.

Judicata (judicata.com) es una herramienta que permite, a partir de datos no estructurados de jurisprudencia, pasar a datos estructurados, ayudando así a organizar grandes cantidades de información, permitiendo una mejor toma de decisiones jurídicas.

eBrevia (ebrevia.com) automatiza el proceso de revisión de contratos jurídicos utilizando aprendizaje automático.

Kira DiligenceEngine (kirasystems.com) automatiza la extracción y el análisis de las disposiciones de los contratos judiciales haciendo posible crear gráficos y resúmenes personalizados.

Por último, **Brightleaf** (brightleaf.com) permite automatizar todo el proceso de extracción de la información de los contratos judiciales. Es capaz de indexar los términos principales y las disposiciones, por ello es un motor de búsqueda bastante rápido. Esto hace posible la búsqueda de información clave dentro de cientos o miles de contratos.

Como se puede observar, ninguna de las aplicaciones comentadas anteriormente está especializada en la jurisprudencia española, lo cual hace del objetivo principal del Trabajo Fin Grado un proyecto aún más interesante, al especializarse en sentencias de ámbito nacional.

Capítulo 2. Desarrollo del proyecto

Las tareas que se abordan en este trabajo requieren la extracción de información y clasificación de las sentencias judiciales. En este capítulo se exponen las fases que se abordaron hasta conseguir el propósito. En primer lugar fue necesario determinar los textos que se abordarían, y comprensión de la estructura de los mismos, lo que se resume en el epígrafe 2.1. A continuación se implementó la funcionalidad de aprendizaje dirigida a la creación de una base de conocimiento. Mediante este recurso se generan los documentos representantes imprescindibles para la clasificación de las nuevas sentencias entrantes. A partir de aquí, se extrae la información estructurada de dichas sentencias (sede, ponente) y la semi-estructurada (juzgado de origen, letrados, procuradores). Por último se construyen los gráficos que permiten visualizar los datos que nos interesan.

2.1 Sentencias judiciales

2.1.1 CENDOJ

El Centro de Documentación Judicial (CENDOJ) es el órgano técnico del Consejo General del Poder Judicial que se ocupa de la publicación oficial de la jurisprudencia, así como de las demás competencias en el ámbito de la documentación y de los servicios de gestión del conocimiento.

En su labor de difusión de la jurisprudencia española, el CENDOJ pone a disposición del público en general, de forma gratuita y a través de la página web del Poder Judicial, las resoluciones judiciales dictadas por los Tribunales españoles.

Ofrece servicios de apoyo e información a los miembros de la Carrera Judicial. Para ello facilita el acceso a todo tipo de fuentes documentales (jurisprudencia, legislación y publicaciones) que son utilizados en el desarrollo

de la actividad judicial, presta servicios de búsqueda de documentación y se ocupa de la red de bibliotecas judiciales.

A continuación se muestra una figura que representa el organigrama de los juzgados y tribunales españoles, ordenados jerárquicamente de mayor a menor rango.

Tribunal Supremo
Sala Primera Civil – Sala Segunda Penal – Sala Tercera Contencioso Administrativo – Sala Cuarta de lo Social – Sala Quinta de lo Militar
Audiencia Nacional
Sala Penal – Sala Social – Sala Contencioso Administrativo
Tribunal Superior de Justicia
Sala Civil y Penal - Sala Social – Sala Contencioso Administrativo
Audiencia Provincial
Sección Civil y Penal
Juzgados
Juzgado Central de Instrucción – Juzgado Central de lo Penal
Juzgado Central de lo Contencioso Administrativo – Juzgado Central de Menores
Primera Instancia – Instrucción – Menores – Vigilancia Penitenciaria de lo Contencioso Administrativo

Tabla 1.Organigrama juzgados y tribunales.

Las sentencias tratadas son aquellas que resuelve la Audiencia Provincial, enviadas de la instancia inferior, en este caso, los juzgados de las diferentes provincias de España.

Las sentencias que resuelve la Audiencia Provincial son recursos a otras sentencias dictaminadas por jueces de los juzgados de menor rango (juzgados de instrucción, juzgados de primera instancia, etc.).

2.1.2 Sentencias tratadas

Las sentencias judiciales son descargadas del CENDOJ en formato PDF. No siguen un formato específico a la hora de ser redactadas, en cambio la estructura que divide las distintas partes de la misma si son similares.

En primer lugar nos encontramos con la información general de la sentencia. En ella consta:

Id Cendoj: Id único de cada sentencia judicial

Órgano: Tipo de órgano que tramita la sentencia

Sede: Lugar de la sede donde se produjo

Sección: Número de la sección

Nº de recurso y **nº de resolución**

Procedimiento: Tipo de procedimiento

Ponente: Magistrado que lleva el caso

Tipo de resolución: Auto, sentencia, acuerdo, etc.

La estructura de la sentencia prosigue con información relativa al presidente, magistrados, lugar y fecha, juzgado de origen (tipo, número y lugar), parte demandante (procurador y letrado), parte demandada (procurador y letrado). Esta información no está estructurada, y según la sentencia puede estar en un único párrafo o estar en varios antes de exponer los antecedentes de hecho.

De esta última parte nos interesa extraer el juzgado de origen, su tipo, número de juzgado y lugar.

El objetivo de la extracción de la información más básica del documento es la de tener los textos judiciales organizados por sedes y ponentes que llevan cada sentencia. Además, conocer el juzgado de donde procede puede ser interesante para determinar la eficacia de la justicia en dichos juzgados (ver Figura 1).

Las partes que prosiguen en la sentencia son los Antecedentes de Hecho y los Fundamentos de Derecho o Fundamentos Jurídicos. Estos puntos no son considerados para la extracción de la información.

Finalmente, se expone el Fallo o Parte resolutiva en la Figura 2.

Consejo General del Poder Judicial | BUSCADOR JURISPRUDENCIA

Roj: SAP TF 227/2015 - ECLI:ES:APTF:2015:227
Id Cendoj: 38038370032015100035
Órgano: Audiencia Provincial
Sede: Santa Cruz de Tenerife
Sección: 3
Nº de Recurso: 545/2014
Nº de Resolución: 37/2015
Procedimiento: Recurso de Apelación
Ponente: MODESTO VALENTIN ADOLFO FERNANDEZ DEL VISO BLANCO
Tipo de Resolución: Sentencia

SENTENCIA

Ilmos. Sres.

Presidente:

D. MODESTO FERNÁNDEZ DEL VISO BLANCO

Magistradas:

Dª. MACARENA GONZÁLEZ DELGADO

Dª. MARÍA DEL CARMEN PADILLA MÁRQUEZ

En Santa Cruz de Tenerife, a diez de febrero de dos mil quince.

Visto por los Ilmos. Sres. Magistrados arriba expresados el presente recurso de apelación interpuesto por la parte demandante, contra la sentencia dictada en los autos de Juicio Verbal nº 252/2013, seguidos ante el Juzgado de Primera Instancia nº 3 de La Orotava, promovidos por D. Alexander , representado por el Procurador D. Juan Pedro González Martín, y asistido por el Letrado D. Justo Clemente Pliego, contra Dª. Adoracion , representada por la Procuradora Dª. María Yurena Sicilia Socas, y asistida por el Letrado D. Antonio Darias Padrón; han pronunciado, en nombre de S.M. EL REY, la presente sentencia:

Figura 1. Información general.

FALLO

1. Estimar el recurso de apelación interpuesto por la representación procesal de don Alexander , contra la sentencia dictada en el presente procedimiento, y revocar la resolución apelada.

2. Estimar en su integridad la demanda formulada contra doña Adoracion , declarando haber lugar al desahucio por precario de la vivienda descrita en la demanda, sita en la CALLE000 nº NUM000 (antes NUM001) en el término municipal de La Victoria de Acentejo, a la que se contrae el procedimiento, condenando a la demandada a dejarla libre y a disposición del actor, con apercibimiento de lanzamiento si no lo verifica en el término legal.

3. Disponer en cuanto a costas lo consignado en el fundamento cuarto.

Contra esta sentencia cabe recurso de casación por interés casacional (art. 477.2.3º de la Ley de Enjuiciamiento Civil), y recurso extraordinario por infracción procesal si se formula conjuntamente con aquél (Disposición Final decimosexta 2ª, de la Ley de Enjuiciamiento Civil), que podrán interponerse ante esta Sala en el plazo de veinte días.

Notifíquese esta resolución a las partes en la forma que determina el artículo 248-4 de la Ley Orgánica del Poder Judicial .

Una Vez sea firme la anterior resolución, devuélvanse los autos originales al Juzgado de procedencia con testimonio de la presente resolución, para su ejecución y cumplimiento y demás efectos legales.

Así por esta, nuestra sentencia, definitivamente juzgado en segunda instancia, lo pronunciamos, mandamos y firmamos.

Figura 2. Fallo o Parte resolutiva.

2.2 Expresiones regulares

Una expresión regular es una secuencia de caracteres que forma un patrón de búsqueda. Esta puede ser usada para la búsqueda de patrones de cadenas u operaciones de sustitución.

Las expresiones regulares se construyen utilizando operaciones de unión, concatenación y clausura de Kleene.

Éstas son de gran utilidad en este proyecto, ya que son usadas en las distintas secciones para detectar fragmentos de texto de los corpus legales, y así poder obtener la información necesaria (ver Figura 3).

```
p = Pattern.compile("(Procuradora?\\s*)(.*?)((\\sy\\s)|[,;])((.*)*)");
p = Pattern.compile("(Letrado?a?\\s*)(.*?)((\\sy\\s)|[,;])((.*)*)");
```

Figura 3. Ejemplo de uso de expresiones regulares.

2.3 Base de conocimiento

Para la creación de la base de conocimiento se requiere tener una cantidad de sentencias previamente clasificadas. La aplicación extrae el fallo(parte resolutiva) y se comprueba cuáles de estas palabras nos aporta un valor en cuanto a su significado, por ello es necesario una lista de palabras vacías

(preposiciones, conjunciones, pronombres). Si la palabra aparece en esta lista, se elimina y no pasa a ser parte de la lista de palabras con frecuencias.

A continuación se realiza el análisis morfológico para recoger el lema de cada palabra. De esta forma, las flexiones de una determinada forma o lema serán considerados como instancias de una misma palabra.

2.3.1 Lematización

El tratamiento de palabras que son flexiones de un mismo lema es el primer problema con el que nos encontramos. Estas palabras representan un significado único, pero difieren en el género, número, conjugación, etc.

Un ejemplo de ello sería:

desestimo – desestimamos – desestima

Las cuales son instancias de un mismo concepto, por lo que deberían ser identificadas, en este caso, como '**desestimar**'.

O por ejemplo:

admitimos – admite – admitido

Las cuales son instancias de un mismo concepto, por lo que serán identificadas como: '**admitir**'.

2.4 Similitud entre documentos

Por cada clase de la base de conocimiento (Favorables, Desfavorables y Parciales) se construye un documento representante, el cual va a servir para determinar la clase de una nueva sentencia que se quiere clasificar.

La similitud entre los documentos representantes y la sentencia a clasificar se realiza utilizando la distancia coseno. Esta viene determinada por:

$$sim(d_1, d_2) = cos(\widehat{d_1, d_2}) = \frac{\langle d_1, d_2 \rangle}{\|d_1\| * \|d_2\|}$$

Siendo d_1, d_2 los dos documentos a comparar,$\langle d_1, d_2 \rangle$ el producto escalar entre dos vectores, n el número de palabras a considerar en los documentos:

$$\langle d_1, d_2 \rangle = x_{11}x_{21} + x_{12}x_{22} + \cdots x_{1n}x_{2n}$$

Y$\|d\|$ la norma o módulo del vector *d*, que viene dada por:

$$\|d\| = \sqrt{x_1^2 + x_2^2 + \cdots x_n^2}$$

El $cos(\widehat{d_1, d_2}) = 1$ si los documentos son exactamente iguales, mientras que $cos(\widehat{d_1, d_2}) = 0$si son totalmente distintos.

En la Tabla 2se presenta un ejemplo de matriz de documentos, en la cual se representa la frecuencia con que aparece cada una de las palabras consideradas en el documento correspondiente. En el ejemplo sólo se han considerado 2 casos.

	desestimar	**imposición**	**integridad**	**proceder**
d1	5	Parte II.	2	3
d2	3	4	Parte III.	1

Tabla 2. Matriz de frecuencias.

Después de aplicar la distancia coseno, observamos la matriz de distanciasen la Tabla 3.

	d1	**d2**
d1	1	0.57
d2	0.57	1

Tabla 3. Matriz de distancias.

Como se puede observar, el documento 1 y el documento 2 obtienen una distancia coseno entre ellos de 0.57.

2.1 Similitud de cadenas

Los juzgados de los que provienen las sentencias originales es frecuente encontrarlos registrados abreviados de distintas formas. Se hace necesario por tanto aplicar técnicas que permitan identificarlos como un mismo juzgado, con este propósito se optó por utilizar la distancia de Levenshtein.

2.1.1 Distancia Levenshtein

La similitud entre dos cadenas de texto A y B se basa en el conjunto mínimo de operaciones de edición necesarias para transformar A en B, o viceversa. Hay tres operaciones de edición, las cuales son destrucción, inserción y substitución.

Por ejemplo la distancia Levenshtein entre las palabras "desestimo" y "desestimar" es de 2 ya que se necesitan al menos 2 ediciones elementales para transformar el uno en el otro:

desestimo→*desestima* (Sustitución de la –o- por –a-)

desestima→*desestimar* (Inserción de la –r-)

2.2 Tecnologías utilizadas

Para definir las tecnologías utilizadas hay que diferenciar dos vertientes del proyecto:

- **Aplicación de extracción de la información**: Esta aplicación ha sido desarrollada usando Java y librerías complementarias para el tratamiento de archivos en formato PDF y procesamiento del lenguaje natural. Se ha elegido Java como lenguaje de programación ya que existen múltiples librerías para la extracción de texto de archivos PDF y la manipulación de los mismos.

  Además de esto, se buscaba utilizar una librería de análisis morfológico para la lematización, para lo que fue elegida la librería FreeLing. Uno de los puntos a su favor es que aún estando programada en C++, puede utilizarse en Java. Su mayor ventaja son los servicios lingüísticos que ofrece para el lenguaje español.

- **Aplicación de visualización**: Esta parte del proyecto ha sido desarrollada utilizando HTML5, CSS3, Javascript y D3.js. Era necesaria la utilización de una librería para manipular los datos recogidos de la anterior aplicación, por ello D3.js fue la elegida para este proyecto. D3 combina componentes de visualización de gran alcance y su enfoque de manipulación de datos.

2.2.1 Java

Java es un lenguaje de programación orientado a objetosque se popularizó a partir del lanzamiento de su primera versión comercial de amplia difusión, la JDK 1.0 en 1996. Actualmente es uno de los lenguajes más usados para la programación en todo el mundo.

El lenguaje de programación fue desarrollado por James Gosling de Sun Microsystems como un componente fundamental de la plataforma Java de SunMicroSystems.

Java es un lenguaje independiente de la plataforma en que se utilice. El código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Las aplicaciones de Java son compiladas de forma que pueden ejecutarse en cualquier máquina virtual Java sin importar la arquitectura de la computadora.

Su sintaxis deriva en gran medida de C y C++ aunque sus utilidades a bajo nivel son menores.

2.2.2 Apache PdfBox

PdfBox es una herramienta de código abierto de Java para trabajar con documentos PDF. Publicado bajo la licencia Apache v2.0 PdfBox permite:

- Extraer texto de archivos PDF.
- Dividir o combinar archivos PDF.
- Extraer datos de formularios o rellenar un formulario PDF.
- Validar archivos PDF con el estándar PDF/A-1b.
- Imprimir archivos PDF usando la API de impresión estándar de Java.
- Guardar archivos PDF como archivos de imagen.
- Crear un archivo PDF desde cero, con fuentes e imágenes incrustadas.

- Firmar digitalmente archivos PDF.

2.2.3 FreeLing

FreeLing es una librería de código abierto desarrollada en el Centro de Investigación TALP (*Center forLanguage and Speech Technologies and Applications*) de la Universidad Politécnica de Cataluña para el procesamiento multilingüe automático, que proporciona una amplia gama de servicios de análisis lingüístico para diversos idiomas. FreeLing ofrece a los desarrolladores de aplicaciones de Procesamiento del Lenguaje Natural funciones de análisis y anotación lingüística de textos, con la consiguiente reducción del coste de construcción de dichas aplicaciones.

El proyecto se estructura como una librería que puede ser llamada desde cualquier aplicación de usuario que requiera servicios de análisis del lenguaje.

La versión actual soporta (a diferentes niveles de completitud) las siguientes lenguas: asturiano, catalán, castellano, galés, gallego, inglés, italiano, portugués, y ruso (ver Tabla 4).

	as	**ca**	**cy**	**en**	**es**	**gl**	**It**	**pt**	**ru**
Tokenization	X	X	X	X	X	X	X	X	X
Sentencesplitting	X	X	X	X	X	X	X	X	X
Numberdetection		X	Part	X	X	X	X	X	X
Date detection		X	Part	X	X	X	Part	X	X
Morphologicaldictionary	X	X	X	X	X	X	X	X	X
Affix rules	X	X	X	X	X	X	X	X	Part
Multiworddetection	X	X	X	X	X	X	X	X	Part
Basic namedentitydetection	X	X	X	X	X	X	X	X	X
B-I-O named entity detection				X	X	X	Part	Part	Part
NamedEntityClassification				X	X	Parte XI	Part	Part	Part
Quantitydetection		X	Part	X	X	X	Part	X	X
PoStagging	X	X	X	X	X	X	X	X	X
WN senseannotation		X	Part	X	X	Parte XI	Part	Part	Part
UKB sensedisambiguation		X	Part	X	X	Parte XX	Part	Part	Part
Shallowparsing	X	X	Part	X	X	▪ X	Part	X	Part
Full/dependencyparsing	X	X	Part	X	X	X	Part	Part	Part
Coreferenceresolution					X	Parte XX	Part	Part	Part

Tabla 4. Servicios de análisis disponibles para cada lengua.

Los servicios utilizados en este proyecto para el lenguaje castellano son la tokenización, la división de frases y el diccionario morfológico, el cual es usado para el análisis del texto y la extracción del lema de las palabras.

2.1.1 JSON

JSON es un formato ligero para el intercambio de datos. Una de las ventajas que ofrece sobre XML es la sencillez de escribir un analizador sintáctico de JSON. JSON es un formato de texto que es completamente independiente del lenguaje, pero utiliza convenciones provenientes de C, C++, C#, Java, Javascript, Perl, Python. Estas propiedades hacen de JSON un buen lenguaje para el intercambio de datos.

JSON se basa en dos estructuras (ver Figura 4):

- Colección de pares nombre/valor.
- Lista ordenada de valores.

```
{
  "glossary": {
    "title":"example glossary",
    "GlossDiv": {
      "title":"S",
      "GlossList": {
        "GlossEntry": {
          "ID":"SGML",
          "SortAs":"SGML",
          "GlossTerm":"Standard Generalized Markup Language",
          "Acronym":"SGML",
          "Abbrev":"ISO 8879:1986",
          "GlossDef": {
            "para":"A meta-markup language, used to create
            markup languages such as DocBook.",
            "GlossSeeAlso": [
              "GML",
              "XML"
            ]
          },
          "GlossSee":"markup"
        }
      }
    }
  }
}
```

Figura 4. Ejemplo de formato JSON.

2.1.2 Javascript

Javascript es un lenguaje de programación interpretado cuyo estándar se denomina ECMAScript. Es un lenguaje basado en objetos, basado en prototipos, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente, aunque también existe su forma del lado del servidor.

Javascript se diseñó con una sintaxis similar a la de C y, aunque adopte nombres y convenciones de Java, no están relacionados y tienen semánticas y propósitos diferentes.

Se provee al lenguaje Javascript de una implementación del DocumentObjectModel (DOM) para su interacción con la página web.

Por otro lado, se hace uso de la librería JQuery. Esta librería permite simplificar la manera de interactuar con los documentos HTML, manejar eventos, manipular el árbol DOM y agregar interacción con la técnica AJAX a páginas web.

Con el uso de las funciones propias de JQuery se logran grandes resultados en menos tiempo y espacio. Es por ello que es usado en nuestra aplicación, ya que simplifica el manejo de eventos desde las opciones del filtrado a las gráficas que se visualizan.

2.1.3 D3.js

D3 es una librería de Javascript para la manipulación de documentos basados en datos. D3 construye los gráficos utilizando HTML, SVG y CSS.

Incrustado en una página web HTML, la biblioteca D3.js utiliza funciones predefinidas de Javascript para seleccionar elementos, crear objetos SVG, añadir estilos, transiciones, efectos dinámicos, etc.

Los conjuntos de datos son ligados a los objetos SVG con facilidad utilizando funciones de D3.js, generando diagramas y gráficos a partir de dichos datos.

Los datos que recibe D3.js pueden estar en varios formatos, siendo los más comunes JSON, CSV (*CommaSeparatedValues*) o GeoJSON, aunque, si es

necesario, las funciones de Javascript pueden escribirse para leer otros formatos de datos.

Las siguientes figuras muestran un ejemplo sencillo de la utilización de datos en D3.js.

```
var data = [{name: 'Favorables', value: favorables},
            {name: 'Desfavorables', value: desfavorables},
            {name: 'Parciales', value: parcial}];
```

Figura 5. Datos en D3.js

```
chart.selectAll(".bar")
    .data(data)
  .enter().append("rect")
    .attr("class", "bar")
    .attr("x", function(d) { return x(d.name); })
    .attr("y", function(d) { return y(d.value); })
    .attr("height", function(d) { return height - y(d.value); })
    .attr("width", x.rangeBand())
    .on('mouseover', function(d) {$(this).attr("class", "h-bar") })
    .on('mouseout', function(d) {$(this).attr("class", "bar")});
```

Figura 6. Inserción de datos en D3.js

2.2 Intercambio de datos

Para estructurar la información que tenemos y que posteriormente sea leída y visualizada, generamos un documento con formato JSON. Este documento contiene toda la información necesaria relativa a cada una de las sentencias judiciales (ver Figura 7).

Este documento es almacenado en el servidor para que sea utilizado por la aplicación web, y esta visualice los datos de las sentencias de forma gráfica.

```
"38038370012014100338.pdf": {
   "nº_resolucion":" 350/2014",
   "sede":" Santa Cruz de Tenerife",
   "2_letrado":"Dª Silvia Hernández Delgado",
   "1_letrado":"Dª Raquel Acevedo González",
   "id_cendoj":" 38038370012014100338",
   "tipo_resolucion":" Sentencia",
   "organo":" Audiencia Provincial",
   "seccion":" 1",
   "procedimiento":" Recurso de Apelacion",
   "clasificacion":"desfavorable",
   "presidente":" D. ALVARO GASPAR PARDO DE ANDRADE ",
   "juzgado": {
      "lugar":"san cristobal de la laguna",
      "tipo":"Primera Instancia",
      "numero":"5"
   },
   "nº_recurso":" 471/2013",
   "1_procurador":"D. Nicolás Díaz de Paiz",
   "fichero":"38038370012014100338.pdf",
   "ponente":" ALVARO GASPAR PARDO DE ANDRADE",
   "2_procurador":"Dª Carlota Falcón Lisón",
   "magistrados":" D MARIA PALOMA FERNANDEZ REGUERA D. ANTONIO
   MARIA RODERO GARCIA "
},
```


Figura 7. Información de sentencia en JSON.

2.3 Visualización

La aplicación web recoge los datos del documento JSON, haciendo uso de D3 para la visualización de estos en gráficos.

- **Visualizador de resoluciones**: D3 recoge los datos necesarios para crear los filtros de Sede y Ponentes y visualiza según la selección, las sentencias favorables, parciales y desfavorables.
- **Visualizador de juzgados**: D3 recorre los juzgados de origen de las sentencias, y muestra aquellas donde en la Audiencia Provincial, la resolución ha sido favorable o parcial. Por lo tanto, se están mostrando aquellos juzgados que, en primera instancia, la Justicia falló.

Capítulo 3. Diseño e implementación

Este capítulo tiene como objetivo obtener una visión detallada del proceso que se sigue en la implementación del proyecto.

3.1 Entorno de trabajo

Para el desarrollo del proyecto se ha utilizado NetBeans, un entorno de trabajo utilizado principalmente para el lenguaje de programación Java, aunque también puede utilizarse para otros lenguajes (HMTL5, PHP, C/C++).

El IDE de NetBeans es un IDE de código abierto completamente en Java, por lo que soporta el desarrollo de todos los tipos de aplicación Java.

Toda la documentación de NetBeans se puede encontrar en su sitio oficial, donde también se ofrece soporte para desarrolladores.

3.2 Estructura de alto nivel

Para la utilización de la aplicación es necesario:

- Tener instalada una máquina virtual Java.
- Tener instalada la librería FreeLing.
- Tener almacenados los PDF de las sentencias judiciales que se requieran para la base de conocimiento y los que se quieran clasificar.
- Para la visualización es necesario tener conexión a internet para acceder al servidor donde se encuentra la aplicación web.

El siguiente diagrama trata de ilustrar la arquitectura software que sigue la aplicación. La aplicación Java se encarga de la extracción de la información y de la clasificación de las sentencias, creando la estructura de datos que se

guarda en un fichero JSON. Este fichero JSON se sube al servidor banot utilizado, y este es interpretado por la aplicación web para mostrar a los usuarios la información en gráficos (ver Figura 8).

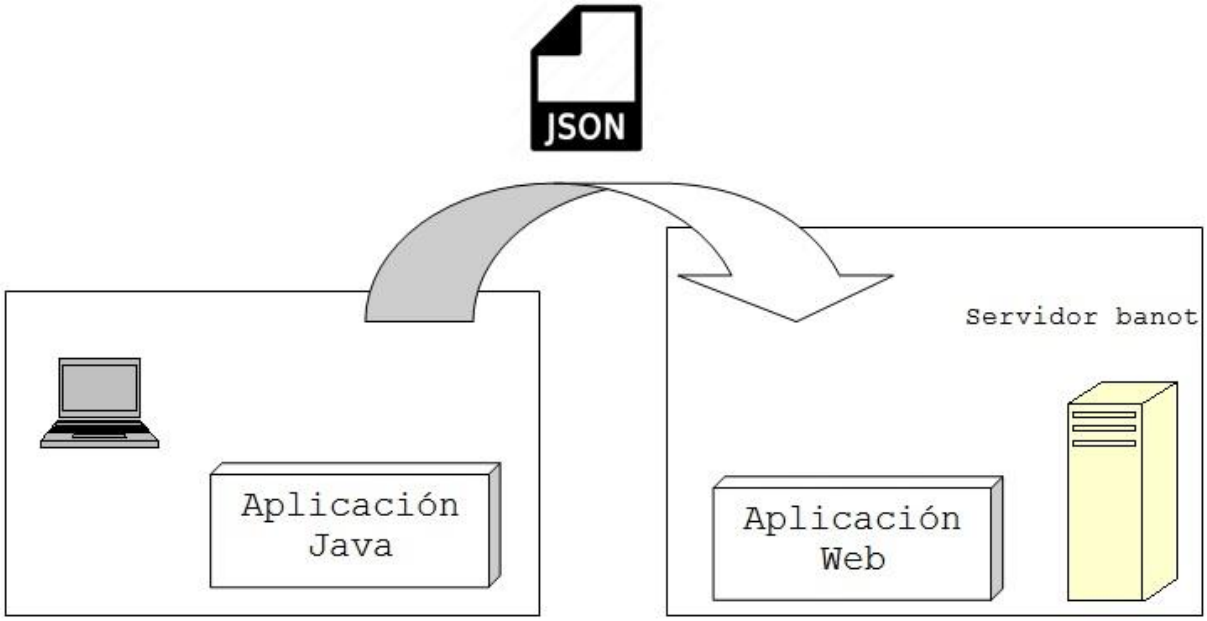

Figura 8. Estructura de la aplicación.

3.3 Base de conocimiento

En esta fase es necesaria la extracción de la parte resolutiva de cada una de las sentencias judiciales previamente clasificadas. Esta extracción se realiza teniendo en cuenta la estructura de la sentencia, ya que la parte resolutiva es la última sección del documento y tiene un título que la representa.

En la extracción del texto de los PDF se nos presenta un problema. Cuando obtenemos el texto utilizando la librería PDFBox anteriormente nombrada, las frases contienen un salto de línea cada vez que se llega al margen del PDF. Por ello, el reconocimiento de los párrafos se hace más difícil de obtener, así como los cortes que creaba en medio de las cadenas de palabra.

Para solucionar dicho problema fue necesaria la creación de un algoritmo que localizara aquellos finales de frase que no acabaran en un párrafo, estableciendo así, las cadenas de palabras que pertenecían al mismo párrafo.

Una vez obtenida la parte resolutiva es necesaria la inicialización de la clase que maneja la carga de la librería FreeLing.

La instalación de la librería FreeLing es otro de los problemas con los que nos encontramos. FreeLing fue instalado en un sistema operativo Ubuntu 14.04.2 (32 bits). Para su correcta ejecución es necesaria la instalación de laslibrerías adicionales, ya que en ausencia de estas librerías, podría dar problemas en cada paso de la instalación. En el caso de este proyecto, nos encontramos con problemas derivados de la instalación de librerías y paquetes adicionales y de su compatibilidad con el sistema operativo que usamos. Dichas librerías son:

- Boost de C++. (Libboost y boost_thread).
- Cabeceras de zlib. (zlib.h en el paquete zlib-dev).

La guía de instalación de FreeLing no proporciona la solución a algunos de los errores que nos producían, por ello fue necesario la búsqueda de información sobre dichos paquetes y su correcta instalación.

La instalación de FreeLing en dicho sistema operativo y algunos de los problemas que se pueden encontrar, así como la solución aplicada se encuentran en el Apéndice A.1.

La clase de la librería FreeLing usada se encarga de cargar los ficheros de datos necesarios para el lenguaje, en nuestro caso español. Locuciones, cuantificadores o abreviaciones son ejemplos de estos ficheros de datos.

A lo largo del proyecto, cuando se crea la base de conocimiento y cuando se clasifican los documentos, es necesaria la obtención del lema de las palabras. Se utiliza FreeLing para realizar un análisis morfológico de las frases que se requiera, y además, descartando las palabras que se encuentren en la lista de palabras vacías.

En este punto, para cada clase de la clasificación: favorable, desfavorable y parcial, se debe crear una matriz de frecuencias de los lemas generados con

FreeLing y la ‘cantidad’ de dichas palabras (Figura 9). De esta forma se obtiene el documento que representará a cada una de las clases.

```
Palabra: <alzada> Frec: <15>
Palabra: <anterior> Frec: <23>
Palabra: <apelación> Frec: <36>
Palabra: <apelante> Frec: <15>
Palabra: <apelar> Frec: <5>
Palabra: <arona> Frec: <3>
Palabra: <art.> Frec: <39>
Palabra: <artículo> Frec: <18>
Palabra: <así> Frec: <40>
Palabra: <audiencia> Frec: <18>
Palabra: <auto> Frec: <48>
Palabra: <autos_de_juicio_cambiario> Frec: <1>
Palabra: <autos_de_juicio_ordinario> Frec: <1>
Palabra: <autos_de_juicio_verbal> Frec: <1>
Palabra: <bajo> Frec: <1>
Palabra: <banco_santander> Frec: <1>
Palabra: <brigida> Frec: <1>
Palabra: <caber> Frec: <41>
Palabra: <casacional> Frec: <39>
Palabra: <casación> Frec: <39>
Palabra: <caso> Frec: <3>
Palabra: <causar> Frec: <1>
Palabra: <certificación> Frec: <10>
Palabra: <certificar> Frec: <18>
Palabra: <citar> Frec: <1>
Palabra: <común> Frec: <1>
Palabra: <condenar> Frec: <3>
Palabra: <confirmar> Frec: <34>
Palabra: <conforme> Frec: <1>
Palabra: <conformidad> Frec: <35>
Palabra: <conjuntamente> Frec: <39>
Palabra: <consecuencia> Frec: <4>
Palabra: <constituir> Frec: <36>
Palabra: <corresponder> Frec: <1>
Palabra: <cosa> Frec: <2>
Palabra: <costa> Frec: <42>
```

Figura 9. Lista de frecuencia de palabras.

3.4 Similitud de documentos

El proceso de clasificación de los nuevos documentos judiciales es similar al nombrado anteriormente, ya que se extrae cada parte resolutiva y se crea la matriz de frecuencias.

En esta fase tendríamos el documento resultante de la sentencia a clasificar, y los tres documentos representantes de cada clase. Finalmente se ha de comparar con cada uno de los tres documentos por similitud de documentos.

La similitud entre los documentos representantes y la sentencia a clasificar se realiza utilizando la distancia coseno.

Es necesaria la creación de una matriz de frecuencias de palabras por documentos (Tabla 2. Matriz de frecuencias). Esta matriz se crea incluyendo 4 documentos:

- La sentencia a clasificar
- El documento representante de la clase favorable
- El documento representante de la clase parcial
- El documento representante de la clase desfavorable

A continuación se aplica la distancia coseno utilizando la matriz de frecuencia de palabras, creando así la matriz de distancias. En la Figura 10se observa un ejemplo de las sentencias clasificadas según su distancia coseno con los documentos representantes de cada clase.

```
Fichero: 38038370012014100389.pdf
Clase: <DESFAVORABLE> | cosenoPar:0.8987262855604634 - cosenoFav: 0.8485093488927226 - cosenoDes: 0.9115664193299523
 ---
Fichero: 28079370242015100048.pdf
Clase: <DESFAVORABLE> | cosenoPar:0.4835233657918853 - cosenoFav: 0.49976959639424445 - cosenoDes: 0.5078085299386796
 ---
Fichero: 28079370222015100202.pdf
Clase: <DESFAVORABLE> | cosenoPar:0.5106114003913735 - cosenoFav: 0.5511283776645073 - cosenoDes: 0.5593573754045307
 ---
Fichero: 28079370202015100092.pdf
Clase: <DESFAVORABLE> | cosenoPar:0.46070423210717937 - cosenoFav: 0.48684078171312106 - cosenoDes: 0.49437915939573374
 ---
Fichero: 38038370012014100327.pdf
Clase: <DESFAVORABLE> | cosenoPar:0.8688560351841804 - cosenoFav: 0.812745132137294 - cosenoDes: 0.8831319381692033
 ---
Fichero: 38038370012014100378.pdf
Clase: <FAVORABLE> | cosenoPar:0.8762510658989952 - cosenoFav: 0.909432420971465 - cosenoDes: 0.8413584726145855
 ---
Fichero: 38038370012014100381.pdf
Clase: <DESFAVORABLE> | cosenoPar:0.8896936313305618 - cosenoFav: 0.8335693285450778 - cosenoDes: 0.9029012240383927
 ---
```

Figura 10. Distancia coseno entre documentos.

3.5 Extracción de figuras judiciales

Las figuras judiciales tratadas en este punto (Procuradores y Letrados) son aquellas que representan a la parte demandante y a la parte demandada.

Estas figuras se redactan entre la información básica y los antecedentes de hecho, haciendo posible localizarla búsqueda tan solo en dicho fragmento con

objeto de optimizar el rendimiento. Además de esto, las figuras son nombradas en el siguiente orden:

- Procurador/a de la parte demandante.
- Letrado/a de la parte demandante.
- Procurador/a de la parte demandada.
- Letrado/a de la parte demandada.

Así, obtenemos los nombres de estas figuras que posteriormente se guardarán en un objeto JSON.

3.6 Extracción del juzgado

El juzgado de origen (juzgado donde procede la sentencia judicial) se puede encontrar en varias partes del documento, aunque siempre después de la información básica y antes de los antecedentes de hecho. Es por ello que es buscado en ese fragmento del documento.

Aquí es necesario diferenciar cierta información relativa al juzgado:

- **Tipo**: El tipo indica si el juzgado es de primera instancia, de instancia e instrucción, etc. Estos tipos se registran en una lista, para aquellas subcadenas que son juzgados se determina la similitud con los tipos en la lista. De esta forma se consigue estandarizar la denominación para todos los juzgados, visualizándolos correctamente.

  (“1”, “Primera”) (“Inst”, “Instancia”)

  (“1ª”, “Primera”) (“Inst.”, “Instancia”)

- **Número**: Para la extracción correcta del número primero es necesario conocer si está escrito con letras o directamente con números, por lo que también se realiza una comparación con una lista de números.

  (“1”, “UNO”)

  (“2”, “DOS”)

  (“3”, “TRES”)

En el caso de que el número ya esté escrito con letras, se realiza una similitud de cadenas utilizando la distancia Levenshtein.

- **Lugar**: El lugar del juzgado viene determinado por la sede de la sentencia judicial. Esto es, dependiendo de la sede, habrá unos posibles juzgados u otros. También se utiliza la distancia Levenshtein para determinar el lugar.

  Por ejemplo: Si llega una sentencia con sede en Santa Cruz de Tenerife se compara el lugar extraído con los juzgados de Santa Cruz de Tenerife, utilizando la distancia Levenshtein, devolviendo el juzgado que más se le parezca.

3.7 Visualización gráfica

Los datos son recogidos por D3 del archivo JSON que se construyó en la aplicación de extracción de información.

Se crean dos tipos de gráficos:

- Un **gráfico de barras** para representar las sentencias con una resolución favorable, desfavorable o parcial. A este gráfico se le puede aplicar un filtro por sedes, y también por los ponentes (ver Figura 11).
- Un **gráfico circular** que muestra los juzgados de origen en los cuales se determinó una sentencia como desfavorable pero en la Audiencia provincial se dio como favorable. Es decir, se representan los casos en los que hay discrepancia entre la sentencia original y el fallo en la Audiencia Provincial. Este gráfico está sujeto a un filtro por sedes (ver Figura 12).

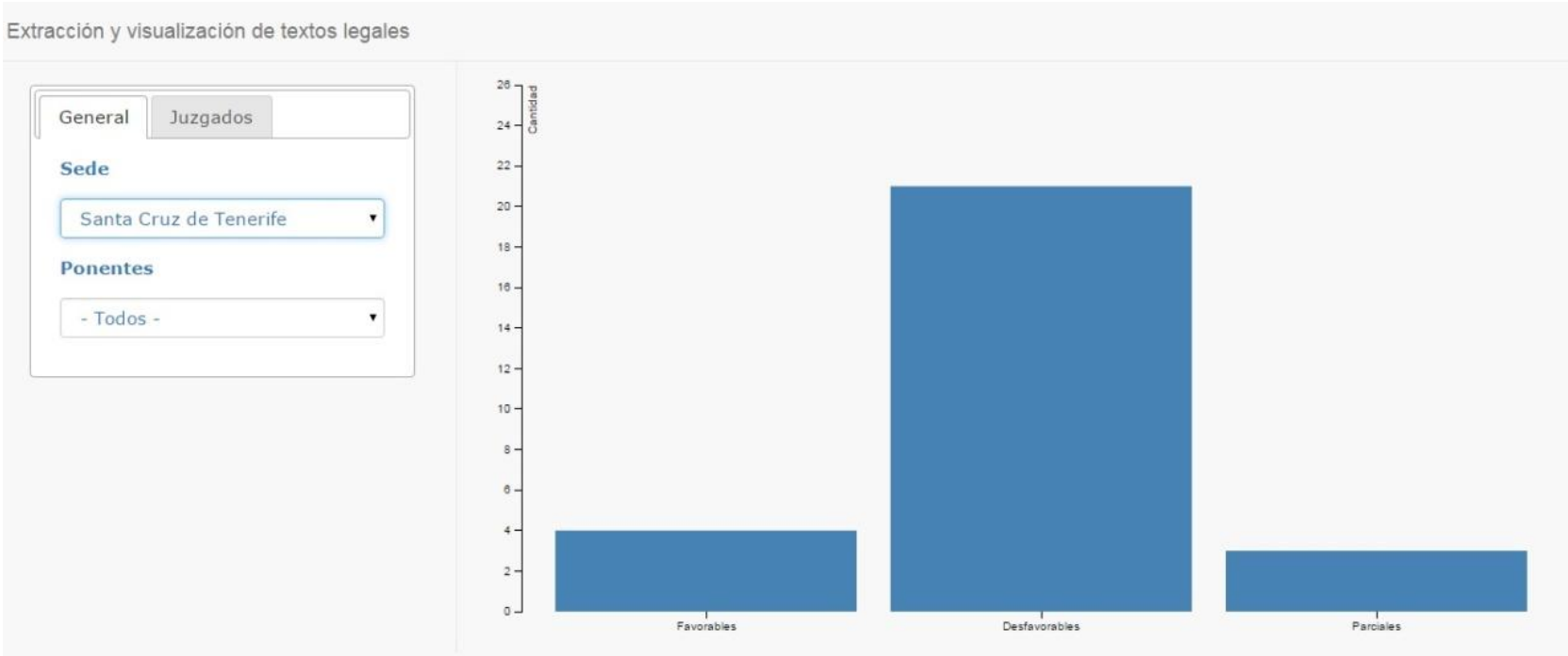

Figura 11. Gráfico de barras.

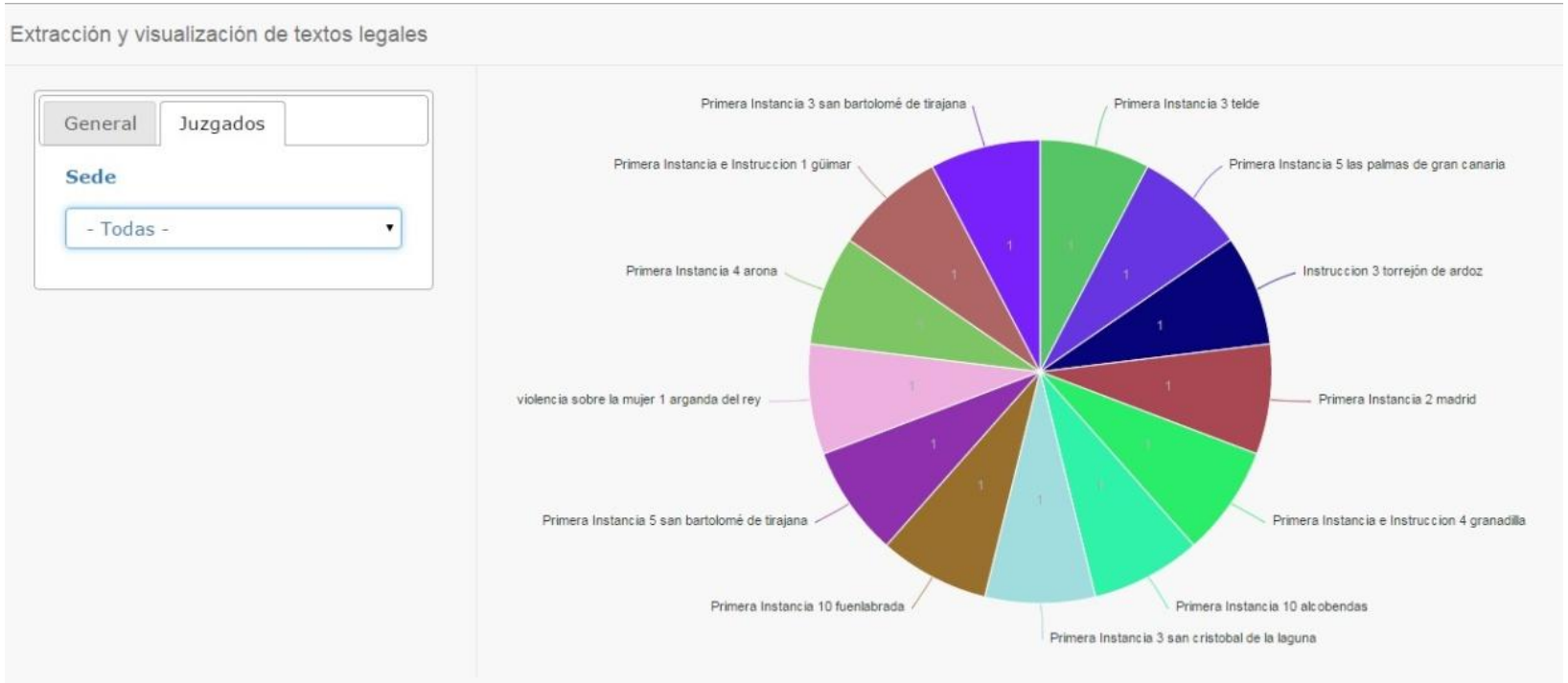

Figura 12. Gráfico Circular.

Capítulo 4. Conclusiones y líneas futuras

4.1 Objetivos conseguidos

En el desarrollo del proyecto se ha conseguido extraer información de las sentencias judiciales en el ámbito propuesto, además de obtener un método clasificador de sentencias.

Se ha logrado obtener los datos que representan información relevante, así como la interpretación de los mismos. Utilizando datos del clasificador y de la extracción del juzgado de origen se han determinado aquellos juzgados donde en primera instancia la sentencia fue desfavorable pero en una instancia superior no lo fue.

Además, el proyecto no sólo se limita a la extracción de la información y al clasificador de sentencias, sino que aporta una interfaz gráfica para la visualización de los datos, lo cual es un punto clave para obtener conclusiones.

4.2 Trabajos futuros

A partir de los resultados obtenidos en este proyecto, se pueden realizar diversos trabajos futuros:

- Dotar a la aplicación de más opciones de visualización, utilizando la mayor parte de los datos que se extraen.
- Extraer información adicional que pueda resultar interesante o relevante (situación familiar, situación laboral, etc.).
- Ampliar el ámbito de análisis de sentencias. Tratar sentencias fuera del tema de Divorcio, incluir más sedes de España, tratar sentencias de otros órganos judiciales.

Además, la posibilidad de mejorar el clasificador de sentencias, mejorando así los resultados obtenidos.

Capítulo 5. Summary and Conclusions

5.1 Goals reached

During the development of this project it has been possible to extract information from the court judgments on the field established. Furthermore, a method of classification for the sentences has been achieved.

Obtaining data representing relevant information has been also achieved as well as their interpretation. Using data from the classifier and the extraction of data from the origin courts, it has been possible to determine the courts where the sentences were firstly evaluated as unfavorable but in a superior evaluation they were not.

Besides the information presented above, it is important to take into account that the project is not limited to the extraction of information and to the sentences classifier. Apart from that, it also provides a graphic interface for the visualization of data which is the main tool to obtaining important conclusions.

5.2 Future works

Several future works can be developed from the results obtained in this project:

- Providing the application with more visualization options using the biggest amount of extracted data as possible.
- Extracting additional information which could be interesting or relevant (familiar situation, job situation, etc.)
- Expanding the field for the sentences analysis. Dealing with sentences out of the divorce topic, including more courts within Spain and dealing with judgments from other judicial bodies.

Furthermore it is important to consider the possibility of improving the sentences classifier, improving in this way the results obtained.

Capítulo 6. Presupuesto

Este capítulo contiene la información relativa al presupuesto. El presupuesto ha sido calculado contemplando las horas dedicadas y la vida útil del ordenador utilizado. Todo el software utilizado es de licencia libre.

Items realizados	**Horas**	**Presupuesto**
Horas de trabajo	300	300 * 10 = 3000 €
Vida útil material	300	0.12 * 500 = 60 €
Licencias de software	-	0 €
		Total = 3060 €

Tabla 5. Tabla resumen del presupuesto.

Capítulo 7. Apéndice A. Manual de instalación

A.1. Instalación FreeLing

A continuación se muestra una guía de instalación de la librería FreeLing 3.1 en Java. Esta guía ha sido probada en Ubuntu 14.04.2 Desktop LTS(32 bits)

- Actualizar sistema desde 'Actualización de software', luego utilizar comando: `$ sudo apt-getupdate`

- Instalar paquetes requeridos

  ```
  $ sudo apt-get install libboost-regex-dev libicu-dev zlib1g-dev
  $ sudo apt-get install libboost-system-dev libboost-program-options-dev
  ```

- Desempaquetar FreeLing

  ```
  $ tarxzvf freeling-3.1.tar.gz
  $ cd freeling-3.1
  ```

- Configure

  ```
  $ ./configure
  ```

- Problemas con zlib.h

  ```
  checkingforzlib.h... no
  zlib.h not found.
  Make sure zlib headers (package zlib-dev or the like) are installed and can be
  found in a standard path. You may need to set CPPFLAGS to specify search path.
  ```

  Solución: `$ sudo apt-getinstall zlib1g-dev`

- Problemas con boost_thread

```
checking for main in -lboost_thread-gcc-mt... no
boost_thread library not found.
Make sure libboost_thread is installed and can be found in a standard path.
You may need to set LDFLAGS to specify search path.
```

Solución: `$ sudo apt-get installlibboost-thread-dev`

- Compilar

```
$ make
```

```
corrector/dicc2phon-dicc2phon.o: In function `_GLOBAL__sub_I_main':
dicc2phon.cc:(.text.startup+0x2c):        undefined        reference        to
`boost::system::generic_category()'
dicc2phon.cc:(.text.startup+0x36):        undefined        reference        to
`boost::system::generic_category()'
dicc2phon.cc:(.text.startup+0x40):        undefined        reference        to
`boost::system::system_category()'
collect2: error: ld returned 1 exit status
```

Solución: Volver a ejecutar `./configure` con estas variables

```
CXXFLAGS=-lboost_system      CPPFLAGS=-lboost_system      LIBS=-lboost_system
./configure
```

- Makeinstall

```
$ sudo makeinstall
```

- Probar FreeLing

```
$ analyze -f es.cfg
Hola, este es mi primer mensaje.
```

- Instalar Java JDK -> leer fichero freeling-3.1/APIs/java/README

```
$ sudo apt-get install openjdk-7-jdk
```

- Instalar SWIG

```
$ sudo apt-getinstallswig
```

- Editarfichero freeling-3.1/APIs/java/Makefile

```
# prefijo del directorio donde está instalado FreeLing
FREELINGDIR = /usr/local

# directorio donde está instalado swig
SWIGDIR = /usr/share/swig2.0

# directorio donde está instalado la jvm
JAVADIR = /usr/lib/jvm/java-7-openjdk-i386
```

- Añadir '-lboost_system' en la siguiente línea del Makefile:

```
$(GCC) -shared -o libfreeling_javaAPI.so freeling_javaAPI.cxx -lfreeling -
lboost_system -L$(FREELINGDIR)/lib -I$(FREELINGDIR)/include -
I$(JAVADIR)/include -I$(JAVADIR)/include/linux -fPIC
```

- Compilar

```
$ make
```

- Poner las variables de entorno

```
# directorio donde he puesto el freeling
$ export FREELINGDIR=/opt/freeling-3.1

# añadir a LD_LIBRARY_PATH losdirectoriosdondeestán:
# 'libfreeling.so' y 'libfreeling_javaAPI.so'
$ export LD_LIBRARY_PATH=/usr/local/lib:$FREELINGDIR/APIs/java

# ruta del paquete 'freeling.jar'
$ export CLASSPATH=$FREELINGDIR/APIs/java
```

- Compilar y ejecutar el ejemplo de Java

```
$ javac Analyzer.java
```

```
$ javaAnalyzer
Hola, este es un mensaje para el ejemplo de Java.


-------- LANG_IDENT results -----------
Language detected (from first line in text): es
-------- TAGGER results -----------
Holahola I
, , Fc
esteeste PD0MS000
es     ser     VSIP3S0     02604760-v:0.00975876/02745332-v:0.00929988/02620587-
v:0.00868238/02664769-v:0.00805665/02616386-v:0.00750232
un uno DI0MS0
mensajemensaje NCMS000 06253690-n:0.0215432/06598915-n:0.0209718
para para SPS00
elel DA0MS0
ejemploejemplo  NCMS000  05820620-n:0.00707546/05937112-n:0.00658792/05925366-
n:0.00603443/05937524-n:0.00586835/07308889-n:0.00573039/10324851-
n:0.00528231/06880533-n:0.00511763/06672752-n:0.0051067
dede SPS00
Java java NP00G00
. . Fp
```

Capítulo 8. Bibliografía

[1] Página oficial de FreeLing (Descarga e instalación) nlp.lsi.upc.edu/freeling

[2] Instrucciones generales instalación FreeLing: nlp.lsi.upc.edu/freeling/doc/userman/html/node12.html

[3] Página oficial de la librería PDFBoxpdfbox.apache.org/download.cgi

[4] Lex Machinalexmachina.com

[5] Counselyticscounselytics.com

[6] Ebreviaebrevia.com

[7] Judicatawww.judicata.com

[8] Kirasystemskirasystems.com

[9] Brightleafbrightleaf.com

[10] Guía:*Creating animations and transitions with D3* blog.visual.ly/creating-animations-and-transitions-with-d3-js/

[11] Guía gráfico de barras D3 www.saigesp.es/d3js-actualizar-con-transiciones/

[12] Guía gráfico de barras D3:*Let'smake a Bar Chart* bost.ocks.org/mike/bar/1/

[13] Trabajo de Fin de Frado. Aplicación móvil para Firefox OS (Gráficos con D3) github.com/beneharo/pulsec

[14] Levenshtein edit distance ideone.com/oOVWYj

[15] Gráfico circular d3pie d3pie.org

[16] D3.js d3js.org

[17] CENDOJ:Buscador del sistema jurisprudencia www.poderjudicial.es/search/indexAN.jsp