SemOs: Resolución de obstáculos semánticos en castellano, inglés y búlgaro

Resumen de la aplicación:

Esta herramienta permite enriquecer textos mediante la detección de obstáculos semánticos en tres lenguas: castellano, inglés y búlgaro. En concreto, estos obstáculos son: palabras raras, palabras especializadas, palabras polisémicas, palabras largas. Para ello dispone de una utilidad de consola y un servicio web. El proceso es el siguiente: primero, se realiza un preproceso del texto donde se realiza un análisis morfológico, así como una desambiguación de las palabras por el sentido más frecuente; después, se detectan los posibles obstáculos de un texto; y, como resultado, para cada uno de ellos se ofrece su definición, sus sinónimos y su complejidad.

Lenguaje de programación: Java

Entorno Operativo: Linux

Requerimientos:

Java 1.6 o superior
Apache Tomcat 6.0 o superior (opcional)
Ant
500 MB de disco duro libre
300 de RAM o superior

Listado de ficheros fuente:

Directorio ***web/config***
configuration.properties

Directorio ***web/web***
index.jsp
install.jsp

Directorio ***web/web/WEB-INF***
sun-jaxws.xml
web.xml

Directorio ***web/web/META-INF***
context.xml

Directorio ***web/ web/js***
bootstrap.min.js
first.js
jquery-1.8.3.min.js

Directorio ***web/ web/css***
bootstrap.min.css
navbar-fixed-top.css
style.css

Directorio ***web/src/conf/***
MANIFEST.MF

Directorio ***web/src/java/es/ua/dictionary***
CheckBabelNetProp.java
DictionaryService.java
DisambiguationService.java
FirstServiceEvents.java
NormalizeService.java
NormalizeServiceEvents.java
ParamsJSON.java
Word.java

Directorio ***web/src/java/ es/ua/ dictionary /connection***
ConnectionPool.java
ConnectionUtils.java

Directorio ***web/src/java/es/ua/dictionary/servlet***
installFirst.java

Directorio ***core***
core3.0.sh

core3.0.bat
manifest.mf

Directorio ***core/config***
splitter_first.dat

Directorio ***core/src/es/ua/first***
First.java

Directorio ***core/src/es/ua/first/config/***
babelnet.properties
dictionary.properties
Directorio ***core/src/es/ua/first/dictionary***
Dictionary.java
DictionaryException.java
WiktionaryParser.java
Directorio ***core/src/es/ua/first/disambiguation***
DisambiguationException.java
Disambiguatior.java
Disambiguatior2.java
Word.java

Directorio ***core/src/es/ua/first/exception***
BadParamException.java
FirstParserException.java

Directorio ***core/src/es/ua/first/normalization***
Normalizator.java
NormalizeException.java

Instalación:

Descargar el proyecto del repositorio de control de versiones.

Se debe generar los ejecutables de la aplicación utilizando ant: uno para el de aplicación de consola (jar) y otro para la aplicación web (archivo war). Además, es necesario el archivo corpus.zip donde se incluyen recursos necesarios para el funcionamiento de la aplicación. Para su descarga, póngase en contacto con los autores.

Opcionalmente, si se desea instalar la aplicación web, se desplegará en el servidor de aplicaciones (Apache Tomcat) el archivo war. Se debe asegurar que los scripts de la aplicación desplegada tengan permisos de ejecución. Para finalizar esta instalación, se debe rellenar un formulario donde se proporciona la ruta al programa GATE.

Ejecución:

Existen dos opciones de ejecución, por línea de comandos o consumiendo un servicio web.

Si se quiere ejecutar desde línea de comandos, se debe colocar en el directorio donde se encuentre la aplicación y ejecutar el script “core3.0.sh” (en plataformas UNIX) o “core3.bat” (en Windows). Este script se encargará de enriquecer un texto con la detección de obstáculos semánticos y para ello es necesario indicar: el archivo de entrada y de salida, la codificación de caracteres, el idioma del texto y qué obstáculos se quieren procesar. También es recomendable indicar las rutas donde encontrar los recursos que la aplicación necesita (BabelNet y Freeling), así como el método de desambiguación a emplear (por defecto, el sentido más frecuente), la información a recibir (sólo definiciones, sólo sinónimos o ambos), y el número de sentidos a considerar. La salida generada será un archivo con el formato de la plataforma GATE que contendrá el texto original enriquecido: cada obstáculo semántico con la información requerida. Para procesar textos en búlgaro, se espera que el documento de entrada se haya analizado lingüísticamente de manera previa.

Si se quieren consumir los servicios web SOAP es necesario crear un cliente para el servicio web mediante las WSDLs generadas durante la instalación. Para obtener el texto enriquecido con los obstáculos semánticos, dicho cliente consumirá el método “disambiguate” del servicio “Disambiguation”. Deberá indicar el documento de entrada en formato GATE y el idioma del texto. Opcionalmente, podrá indicar: los obstáculos a detectar, el tipo de información a devolver y el número máximo de sentidos a considerar. La salida generada será un archivo con el formato de la plataforma GATE que contendrá el texto original enriquecido para cada obstáculo semántico detectado con la información requerida.

Librerías utilizadas:

Core:

- BabelNet 1.1.0 – Librería para acceder al recurso BabelNet, un diccionario enciclopédico multilingüe.
- XMLUtils 2.0 – Librería que desarrollada por el GPLSI para tratar más fácilmente documentos XML.
- Gate Embedded 7.1 – Infraestructura para desarrollar componentes software que procesan el lenguaje natural.
- CopyLibs – Librería encargada de gestionar otras librerías.
- JWNL14 rc2 – Librería encargada de acceder al recurso Wordnet, un diccionario relacional.
- Wikipedia API 0.9.0 - Librería encargada de acceder al recurso Wictionario

Web

- JSONLib-2.4 y Jackson 3.2 – Librerías encargadas de gestionar objetos en formato JSON
- Metro – Librería encargada de gestionar los mensajes entre servicios web
- Gate Embedded – 7.1 – Infraestructura para desarrollar componentes software que procesan el lenguaje natural.
- CopyLibs – Librería encargada de gestionar otras librerías.
- Commons Logging - Librería de Apache para gestionar los logs de la aplicación
- HttpClient 4.1.2 y HttpCore 4.1.3 – Librerías para construir clientes para peticiones HTTP
- Wikipedia API 0.9.0 - Librería encargada de acceder al recurso Wictionario
- JDBC – Driver para acceder a las diversas bases de datos SQL empleadas
- JWNL14 rc2 – Librería encargada de acceder al recurso Wordnet, un diccionario relacional.

Dependencias de programas de terceros:

- Freeling 3.0 – El analizador lingüístico Freeling debe estar instalado en la máquina y se debe actualizar el fichero freeling.sh (scripts/freeling.sh) con el directorio donde se encuentra esta herramienta.
- Gate Embedded – 7.1 – Esta infraestructura también debe estar instalada en la máquina y se debe indicar su ruta tanto en la aplicación de consola como en la instalación el servicio web.

Dependencias de recursos de terceros:

Existen dos tipos de recursos: los que requieren una base de datos SQL y los que requieren un conjunto de ficheros. En el caso de estos últimos, por defecto, se espera que estos se coloquen en la siguiente ruta relativa (core/corpus/${recurso}); aunque es posible configurar la ruta. En concreto, los recursos de terceros son:

- BalbelNet: Diccionario enciclopédico multilingüe. La ruta donde se encuentra esta herramienta se

debe indicar tanto en la aplicación de consola como en la instalación el servicio web.

- Balkanet WordNet: Base datos léxica de los lenguajes de los Balcanes. La base de datos SQL dónde se encuentra este recurso se debe indicar en el archivo core/src/es/ua/first/config/dictionary.properties.
- Multilingual Central Repository: Base datos léxica del Español, entre otros. La base de datos SQL dónde se encuentra este recurso se debe indicar en el archivo core/src/es/ua/first/config/dictionary.properties.
- Wictionario: Diccionario multilingüe libre para diferentes lenguas, con significados, etimologías y pronunciaciones. La base de datos SQL dónde se encuentra este recurso se debe indicar en el archivo core/src/es/ua/first/config/dictionary.properties.
- WordNet 3.0: Base datos léxica del inglés. La ruta donde se encuentra esta herramienta se debe indicar tanto en la aplicación de consola como en la instalación el servicio web.
- WordNet Domains: Recurso léxico que aumenta de forma semiautomática de WordNet, donde a cada synset se le asigna al menos una etiqueta de dominio. La ruta donde se encuentra esta herramienta se debe indicar en la propiedad *pathWNDomains* del archivo core/src/es/ua/first/config/dictionary.properties.
- WordNet Mappings: Recurso que especifica un mapeo automático entre WordNet 2.0 y 3.0. La ruta donde se encuentra esta herramienta se debe indicar en las propiedad *pathWNMapping** del archivo core/src/es/ua/first/config/dictionary.properties.

Contenido:

core/ - Ficheros de la aplicación java

config/ - Ficheros necesarios para la configuración
lib/ - Librerías usadas.
scripts/ - Ficheros con scripts necesarios para la aplicación
src/ - Los archivos fuentes del back-end.
nbproject/ - Ficheros de configuración necesarios para compilar el proyecto

web/ - Ficheros de la aplicación web

config/ - Ficheros necesarios para la configuración
lib/ - Librerías usadas.
src/ - Los archivos fuentes de la lógica del front-end.
web/ - Los archivo fuentes del front-end
nbproject/ - Ficheros de configuración necesarios para compilar el proyecto