INSTITUTO POLITÉCNICO NACIONAL
CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN
Laboratorio de Lenguaje Natural

Optimización global de coherencia en la desambiguación del sentido de las palabras

T E S I S

QUE PARA OBTENER EL GRADO DE
DOCTOR EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA

M. en C. SULEMA TORRES RAMOS

Director:
Dr. Alexander Gelbukh

México, D.F.
Enero, 2010

INSTITUTO POLITÉCNICO NACIONAL

SIP-14

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

ACTA DE REVISIÓN DE TESIS

En la Ciudad de México, D. F. siendo las 14:00 horas del día 17 del mes de Diciembre de 2009 se reunieron los miembros de la Comisión Revisora de Tesis designada por el Colegio de Profesores de Estudios de Posgrado e Investigación del:

Centro de Investigación en Computación

para examinar la tesis de grado titulada:

"OPTIMIZACIÓN GLOBAL DE COHERENCIA EN LA DESAMBIGUACIÓN DEL SENTIDO DE LAS PALABRAS"

Presentada por el alumno:

TORRES	**RAMOS**	**SULEMA**
Apellido paterno	Materno	nombre(s)

Con registro:

B	0	6	0	9	0	6

aspirante al grado de: ***DOCTORADO EN CIENCIAS DE LA COMPUTACIÓN***

Después de intercambiar opiniones los miembros de la Comisión manifestaron ***SU APROBACIÓN DE LA TESIS***, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Presidente

Dr. Grigori Sidorov

Secretario

Dr. Sergio Suárez Guerra

Primer vocal
(Director de Tesis)

Dr. Alexandre Felixovich Guelboukh Kahn

Segundo vocal

Dra. Sofía Natalia Galicia Haro

Tercer vocal

Dr. Raúl Morales Carrasco

EL PRESIDENTE DEL COLEGIO

Dr. Jaime Álvarez Gallegos

INSTITUTO POLITECNICO NACIONAL
CENTRO DE INVESTIGACION
EN COMPUTACION
DIRECCION

INSTITUTO POLITECNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

CARTA CESION DE DERECHOS

En la Ciudad de México el día 06 del mes de Enero del año 2010, la que suscribe *Sulema Torres Ramos* alumna del Programa de *Doctorado en Ciencias de la Computación* con número de registro *B060906*, adscrita al *Centro de Investigación en Computación*, manifiesta que es autor intelectual del presente trabajo de Tesis bajo la dirección del *Dr. Alexander Gelbukh* y cede los derechos del trabajo intitulado *Optimización global de coherencia en la desambiguación del sentido de las palabras*, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del autor y/o director del trabajo. Este puede ser obtenido escribiendo a la siguiente dirección *sulema7@hotmail.com*. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Sulema Torres Ramos

Nombre y firma

Contenido general de la tesis

Índice detallado de la tesis

Lista de figuras

Lista de tablas

Abstract

Word Sense Disambiguation is the task of choose a sense, from a set of predefined possibilities, for a word in a given text.

Word Sense Disambiguation is considered one of the most important investigation problems in Natural Language Processing. Is very important in applications which need to understand the natural language, like man-machine communication, machine translation, information retrieval, etc.

One of the proposed methods to solve this problem is the Lesk Algorithm. This proposes to use the global text coherence, that is, the total of senses of words related in the text.

The advantage of this method is that we only need one lexical resource, a dictionary of senses. The main disadvantage is that while more words, the search space increase exponentially. So, global optimization methods are used in order to find the optimal combination of senses.

The purpose of this thesis is to improve the results of methods of word sense disambiguation based in direct application of dictionary of senses, using best search methods of optimal combination of senses in a rank of text.

Resumen

La desambiguación del sentido de las palabras es el problema de seleccionar un sentido, de un conjunto de posibilidades predefinidas, para una palabra dada en un texto o discurso.

La desambiguación del sentido de las palabras, es considerada como uno de los problemas más importantes de investigación en el procesamiento del lenguaje natural. Es esencial para las aplicaciones que requieren la comprensión del lenguaje, como la comunicación hombre-máquina, traducción automática, recuperación de la información y otros.

Uno de los métodos propuestos para llevar acabo esta tarea es el método de Lesk, el cual propone utilizar la coherencia global del texto, es decir, el total de sentidos de palabras relacionadas en el texto.

La ventaja de este método es que sólo necesitamos un diccionario de sentidos como recurso léxico. El problema principal es que mientras más palabras tengamos, más grande es el espacio de búsqueda. Por lo tanto, se utilizan métodos de optimización global para buscar la combinación de sentidos cercana al óptimo.

El propósito de esta tesis consiste en mejorar el desempeño de los métodos para la desambiguación de sentidos de palabras basados en la aplicación directa del diccionario de sentidos, a través de aplicación de mejores métodos de búsqueda de combinación óptima de sentidos en un rango de texto.

CAPÍTULO

1

Introducción

Capítulo 1. Introducción

1.1 Ubicación y alcance

La información es el recurso más importante que poseemos los seres humanos. Gran parte de esta información se comunica, almacena y maneja en forma de lenguaje natural (español, inglés, ruso, etc.). En la actualidad, podemos obtener grandes volúmenes de información en forma escrita, ya sea de manera impresa o electrónica.

Las computadoras son una herramienta indispensable para el procesamiento de la información plasmada en los textos, ya que son capaces de manejar grandes volúmenes de datos. Sin embargo, una computadora no puede hacer todo lo que las personas normalmente hacemos con el texto, por ejemplo, responder preguntas basándose en la información proporcionada, o hacer inferencias lógicas sobre su contenido, o elaborar un resumen de esta información.

Por lo anterior, el Procesamiento de Lenguaje Natural (PLN) tiene por objetivo habilitar a las computadoras para que entiendan el texto, procesándolo por su sentido. Para llevar a cabo esta tarea, un sistema de PLN necesita conocer sobre la estructura del lenguaje, la cual se analiza normalmente en 4 niveles: morfológico, sintáctico, semántico y pragmático. En el nivel morfológico se estudia cómo se construyen las palabras; en el sintáctico, cómo combinar las palabras para formar oraciones; en el semántico, el significado de las palabras, y por último, en el pragmático se estudia cómo el contexto afecta a la interpretación de las oraciones. Nuestra investigación se centra en el nivel semántico.

Todos los niveles anteriores de la estructura del lenguaje tienen un problema: la ambigüedad. Ésta se presenta cuando pueden admitirse distintas interpretaciones de un texto, oración o palabra. Resolver la ambigüedad es uno de los principales objetivos del PLN. Existen varios tipos de ambigüedad: sintáctica (estructural), léxica y semántica (ambigüedad del sentido de las palabras).

En el presente trabajo nos centramos en el problema de la ambigüedad del sentido de las palabras, que se presenta cuando una palabra tiene múltiples significados, por ejemplo, el sustantivo "bank" (banco) tiene 10 sentidos (significados) diferentes (según WordNet 2.1, véase la Tabla 1).

Tabla 1. Sentidos obtenidos de WordNet 2.1 para el sustantivo "bank" (banco).

Sentido	Definición
1	A financial institution that accepts deposits and channels the money into lending activities
2	Sloping land (especially the slope beside a body of water)
3	A supply or stock held in reserve for future use (especially in emergencies)
4	A building in which the business of banking transacted
5	An arrangement of similar objects in a row or in tiers
6	A container (usually with a slot in the top) for keeping money at home
7	A long ridge or pile
8	The funds held by a gambling house or the dealer in some gambling games
9	A slope in the turn of a road or track; the outside is higher than the inside in order to reduce the effects of centrifugal force
10	A flight maneuver; aircraft tips laterally about its longitudinal axis (especially in turning

Una forma de resolver la ambigüedad semántica es tomando en cuenta lo coherencia global del texto (Lesk 1986), es decir, el total de sentidos de palabras relacionadas en el texto. La limitación principal de esta técnica es que, para encontrar la combinación óptima de sentidos se necesita mucho tiempo, ya que el espacio de búsqueda es muy grande. Por ello, se usan métodos de optimización global que buscan la combinación óptima de sentidos. Esta tesis está enfocada en obtener mejores resultados en los métodos para desambiguación del sentido de las palabras (WSD), basados en el algoritmo de Lesk, utilizando mejores métodos de optimización global.

1.2 Objetivos

1.2.1 Objetivo general

Mejorar el desempeño de los métodos para la desambiguación del sentido de las palabras basados en la aplicación directa del diccionario de sentidos, a través de aplicación de mejores métodos de búsqueda de combinación óptima de sentidos en un rango de texto.

1.2.2 Objetivos específicos

1. Buscar y/o desarrollar nuevos métodos de optimización global que más se adecuen al problema de WSD basada en Lesk y sus variantes.
2. Identificar ventajas y desventajas de estos métodos
3. Utilizar éstos para la desambiguación del sentido de las palabras
4. Evaluar los resultados obtenidos
5. Modificar el algoritmo original de Lesk para la obtención de mejores resultados.
6. Desarrollo e implementación de una variante del algoritmo de Lesk, modificando su naturaleza lingüística.
7. Evaluación del método modificado de Lesk.

1.3 Relevancia e importancia

Mejorar el desempeño de los métodos para la desambiguación del sentido de las palabras, en este caso específico, el método de Lesk, es esencial para las aplicaciones que requieren la comprensión del lenguaje. Entre estas aplicaciones se encuentran:

a) ***Traducción automática***. La desambiguación semántica es esencial para la traducción apropiada de palabras como *bank* (*banco)* que, dependiendo del contexto, puede traducirse como *institución bancaria*, *orilla del río*, etc. (Weaver 1949) y (Yngve 1955).

b) ***Recuperación de información***. Al realizar búsquedas por palabras claves específicas, es necesario eliminar los documentos donde se usa la palabra o palabras en un sentido diferente al deseado; por ejemplo, al buscar referencias sobre animales con la palabra gato, es deseable eliminar los documentos que contienen dicha palabra asociada con mecánica automotriz (Salton 1968), (Salton & McGill 1983), (Krovetz & Croft 1992), (Voorhees 1993), (Schütze & Pedersen 1995).

c) ***El procesamiento de texto.*** La desambiguación es necesaria para algunas tareas de procesamiento de texto, por ejemplo, para determinar cuándo deben insertarse acentos diacríticos (Yarowsky 1994) y para la detección y corrección del malapropismo (Hirst 1998).

1.4 Novedad científica

Actualmente, cada vez hay más aplicaciones del procesamiento de lenguaje natural que dependen de lo que significa una oración, la cual depende a su vez de lo que significan las palabras en ella.

Tener un método que utilice nuevas técnicas, como las heurísticas modernas de optimización global, que mejoren los resultados de los algoritmos de desambiguación semántica basados en la aplicación directa del diccionario de sentidos, tiene muchas ventajas debido a que las aplicaciones que requieren resolver esta ambigüedad cada vez son mas exigentes y requieren de mejores resultados.

En esta tesis, el desarrollo y/o aplicación de mejores métodos de optimización global en la desambiguación del sentido de las palabras es una novedad, ya que sólo se han aplicado dos de éstos (algoritmos genéticos y templado simulado) para resolver la ambigüedad semántica, no dando muy buenos resultados o siendo costosos en cuestión de tiempo.

Además, se proponen dos métodos basados en los algoritmos tipo Lesk para obtener mejores resultados en la desambiguación del sentido de las palabras.

Otra novedad es que los métodos propuestos en esta tesis son libres del lenguaje, lo cual es muy importante debido a que en todos los lenguajes existe la ambigüedad semántica y se podrían aplicar estos métodos sin necesidad de hacer muchos cambios.

1.5 Aportaciones principales

- Nuevo método para desambiguación del sentido de las palabras, basado en la unión de dos métodos ya existentes (Lesk simple y Lesk completo).
- Nuevo método para desambiguación del sentido de las palabras, basado en una modificación a un algoritmo existente (Lesk simple).
- Demostración de que la aplicación de un mejor método de optimización en el algoritmo original de Lesk, aumenta sus resultados.
- Demostración de que, a diferencia de lo que se ha planteado por otros autores en el estado del arte, el método de Lesk completo con un mejor método de optimización muestra mejores resultados en comparación con los del método simplificado de Lesk.
- Análisis de la cobertura, precisión y *recall* obtenidos para los métodos para la desambiguación del sentido de las palabras basados en la aplicación directa de diccionario de sentidos, como son Lesk completo y Lesk simple; y la interacción que tienen estos algoritmos con diferentes métodos de *back-off*.

1.6 Organización de la tesis

El resto del documento se organiza de la siguiente manera:

En el capítulo 2 "Estado del arte", se presentan los principales métodos tipo Lesk usados para la desambiguación del sentido de las palabras y sus resultados.

En el capítulo 3 "El procesamiento de lenguaje natural y la ambigüedad del sentido de las palabras", se presenta una revisión del estado del arte sobre el lenguaje, procesamiento de lenguaje natural y la ambigüedad, más a detalle, la ambigüedad del sentido de las palabras. También se da una descripción de los métodos para

desambiguación del sentido de las palabras clasificados en base a los recursos que utilizan. Se describe detalladamente el algoritmo de Lesk original y Lesk simple. Se explica cómo se evalúan los métodos para desambiguación del sentido de las palabras y los recursos que utilizan para llevar a cabo esta tarea, así como las medidas de evaluación usadas.

En el capítulo 4 "Métodos de optimización global", se describe la optimización y algunos métodos heurísticos de optimización modernos. Además se explican a detalle algunos de los métodos de optimización global que han sido utilizados para desambiguación del sentido de las palabras. También se describen los algoritmos con estimación de distribuciones que son utilizados en esta tesis para resolver la ambigüedad semántica.

En el capítulo 5 "Propuesta", se describen cuatro propuestas para mejorar el desempeño de los algoritmos tipo Lesk para WSD, específicamente, se propone el uso de algoritmos con estimación de distribuciones para mejorar los resultados de Lesk completo en comparación con otros métodos de optimización, además se propone usar este mismo método de optimización para demostrar que el algoritmo de Lesk completo es mejor que el algoritmo simplificado de Lesk. Se proponen dos métodos nuevos para desambiguación del sentido de las palabras, uno basado en la unión de Lesk simple con Lesk optimizado y otro basado en una modificación al algoritmo de Lesk simplificado.

En el capítulo 6 "Resultados experimentales", se describe la metodología experimental para la evaluación del algoritmo de Lesk y sus variantes. También se explican los parámetros del método de optimización global usado en esta tesis para la desambiguación semántica usando el algoritmo original de Lesk. Se presentan los resultados obtenidos y el análisis de ellos.

En el capítulo 7 "Conclusiones y trabajo futuro", se presentan las conclusiones aportaciones y trabajos publicados derivados de este trabajo, así como el trabajo futuro.

Finalmente se incluyen referencias, ordenadas alfabéticamente y los **anexos**.

CAPÍTULO

2

Estado del arte

Capítulo 2. Estado del arte

2.1 Introducción

En este capítulo se describen los principales métodos que han sido utilizados para la desambiguación del sentido de las palabras. También se presentan los resultados obtenidos para estos métodos.

2.2 Descripción del estado del arte

La primera vez que fue presentada la desambiguación del sentidos de las palabras, como un tarea computacional distinta, fue a finales de los años 40's. En 1949 Weaver introdujo el problema en su ahora famoso memorando en traducción automática, en donde plantea las bases para un nuevo enfoque, la desambiguación del sentido de las palabras (*Word Sense Disambiguation* o WSD por sus siglas en inglés) (Weaver 1949).

El año de 1980 fue un parte aguas en el avance en WSD ya que muchos recursos comenzaron a aparecer disponibles para cualquiera: por ejemplo fuentes léxicas de gran escala, corpus etiquetados a mano (Wilks *et al.* 1990) entre otros. Como consecuencia se fueron desarrollando diferentes algoritmos que utilizaban estas nuevas ventajas, tal es el caso de los algoritmos tipo Lesk, llamados así en honor a su creador.

En el artículo seminal de los algoritmos tipo Lesk (Lesk 1986) se plantea una nueva forma de desambiguación del sentido de las palabras. Se basa en dos ideas principales, un algoritmo de optimización y una medida de similitud para medir la relación entre las definiciones de los sentidos. El algoritmo de optimización considera la coherencia global del texto, esto es, encontrar la combinación de sentidos que maximice la relación total entre los sentidos de todas las palabras.

El problema principal de este algoritmo es la explosión combinatoria que éste representa para un volumen de datos grande, es decir, cuando se tienen varias palabras con varios sentidos cada una.

Por esta razón, dos principales soluciones se han propuesto para aliviar dicha situación,

1) El primero es conocido como algoritmo simplificado de Lesk. En éste, los sentidos de las palabras en el texto son determinados uno por uno encontrando el mayor traslape entre las definiciones de cada palabra y su contexto actual.

2) La aplicación de métodos heurísticos para la obtención del óptimo global en un menor tiempo.

Entre los principales trabajos se encuentra el presentado por Cowie *et al.* (1992) en el que se evalúa el algoritmo original de Lesk usando un algoritmo de optimización conocido como Templado Simulado. Lo evaluó sobre 50 oraciones etiquetadas manualmente y utilizando el diccionario LDOCE (Longman Dictionary of Contemporary English). Las oraciones tenían de dos a quince palabras, con un promedio de 5.5 palabras ambiguas por oración.

Otro de los trabajos relevantes, es precisamente el de la presentación del algoritmo simplificado de Lesk, propuesto por Kilgarrif y Rosenzweig (2000), el se evaluó sobre los datos de SENSEVAL con los sentidos obtenidos del diccionario léxico HECTOR.

Posteriormente, y ya que se conocía el algoritmo de Lesk simple, la pregunta obligada fue, ¿Cuál de estos dos algoritmos obtendría un mejor desempeño? El trabajo presentado por Vasilescu *et al.* (2004) presenta precisamente una comparación entre el algoritmo original de Lesk y el algoritmo de Lesk simple con *back-off* a sentido más frecuente. Dicha comparación fue hecha sobre SENSEVAL-2 English-all words y SemCor.

El mismo año McCarthy *et al.* (2004) reportó resultados sobre SENSEVAL-2 utilizando un algoritmo que deriva, de manera no supervisada, el sentido más frecuente para una palabra utilizando una distribución de similitudes aprendida de un gran número de corpus.

Un año más tarde en el trabajo presentado por Mihalcea (2005), se evalúo el algoritmo original de Lesk con templado simulado y el algoritmo simplificado de Lesk, ambos con *back-off* a sentido aleatorio sobre tres diferentes corpus: SENSEVAL-2, SENSEVAL-3 y una muestra de 10 oraciones seleccionadas aleatoriamente del corpus Semcor.

En la Tabla 2 se presentan los resultados para los trabajos reportados en el estado del arte que llevan a cabo la desambiguación del sentido de las palabras basados en algoritmos tipo Lesk.

Tabla 2. Resultados para algoritmos tipo Lesk reportados en el estado del arte

Referencia	Corpus	Diccionario	Método	Simple	Completo	Sólo back-off
Lesk, 1986	Ejemplos cortos de *Pride and Prejudice and an Associated Press news story*	Oxford Advanced Learner's Dictionary	Exhaustivo	–	50–70	-
Cowie et al., 1992	50 oraciones de LDOCE	LDOCE+	Templado Simulado	–	47	-
Kilgarrif y Rosenzweig, 2000	SENSEVAL	HECTOR lexical database	Glosa y ejemplos	55	–	-
			Sólo glosa	30	–	-
Vasilescu	SENSEVAL-2	WordNet	back-off SMF[2];	55	43	(62.6)

et al., 2004	Semcor (20964 instancias)		exhaustivo para Lesk completo[2]	~55	~43	(73.7)
McCarthy etal, 2004	SENSEVAL-2	WordNet	Sentido más frecuente obtenido automáticamente de corpus no marcados	Precisión	*Recall*	
				53	49	
Mihalcea, 2005	SENSEVAL-2	WordNet	back-off SA[1]; Templado simulado para Lesk completo[1]	48.7	**39.5**	37.9
	SENSEVAL-3		Back-off SA	48.1	–	34.3
	Semcor/10*		Back-off SA	47.4	–	35.3
	Semcor/21*			43.6	–	34.1

+(*Longman Dictionary of Contemporary English)*

* /*n* significa una muestra aleatoria de *n* documentos

[1] Usan traslape como medida de similitud pero lo normalizan (dividiendo entre largo de las definiciones) para evitar la ventaja de definiciones largas. Usan venta de oración

[2] Usan ventana de contexto de 2, 3, 8, 10 y 25

CAPÍTULO

3

El procesamiento del lenguaje natural y la ambigüedad del sentido de las palabras

Capítulo 3. El procesamiento del lenguaje natural y la ambigüedad del sentido de las palabras

3.1 Introducción

La ambigüedad semántica es un problema que se presenta en todos los lenguajes naturales. Podríamos decir que para los seres humanos la ambigüedad en el lenguaje pasa desapercibida, debido a que la resolvemos casi inconcientemente utilizando la realidad en que vivimos, el contexto y el conocimiento que poseemos sobre algunos temas. Pero para las computadoras no es así, por ello el procesamiento de lenguaje natural tiene por objetivo capturar esta información, dando la funcionalidad necesaria a las computadoras para que puedan analizar y procesar lenguaje natural, y así, intentar comprender cómo lo hacemos las personas.

A lo largo de este capítulo daremos una breve revisión del estado del arte sobre el lenguaje y el procesamiento de lenguaje natural. Después hablaremos del problema de la ambigüedad, y más a detalle, la ambigüedad semántica. También veremos la clasificación de los métodos para WSD con base en los recursos que utilizan y explicaremos como funcionan. Más detalladamente, explicaremos el algoritmo original de Lesk y Lesk simple. Se presentan las medidas de similitud propuestas por otros autores para obtener la relación de similitud entre las definiciones de sentidos de las palabras.

Por último se presenta cómo se evalúan los métodos para WSD y se describen los recursos lingüísticos y medidas de evaluación más utilizados para esta tarea.

3.2 Lenguaje

Un lenguaje se considera como un conjunto de frases y oraciones, generalmente infinito y que se forma mediante combinaciones de palabras definidas en un diccionario previamente establecido. Estas combinaciones deben estructurarse correctamente, es decir, respetar un conjunto de reglas sintácticas; además deben tener sentido en un contexto dado, a lo que se denomina semántica.

A lo largo de la historia el ser humano ha utilizado el lenguaje para trasmitir sus conocimientos, sentimientos, emociones, sensaciones, con el fin de comunicarse con el resto de los humanos, ya sea de manera oral, gráfica, escrita o por señas.

Cuando hablamos de lenguajes se pueden diferenciar dos clases muy bien definidas: los lenguajes naturales (español, ruso, inglés, etc.) y los lenguajes formales (lenguajes de programación, lógica matemática, etc.)

3.2.1 Lenguaje natural

Podríamos definir el lenguaje natural como el medio principal para la comunicación humana.

Con respecto a nuestro mundo, el lenguaje nos permite designar las cosas reales y razonar acerca de ellas, así como también crear significados. El lenguaje natural fue desarrollado y organizado a partir de la experiencia humana.

Los lenguajes naturales tienen un gran poder expresivo y pueden ser utilizados para analizar y razonar situaciones complejas.

Una propiedad única de los lenguajes naturales es la polisemia, es decir, la posibilidad de que una palabra en una oración tenga diversos significados (Sidorov 2001). El carácter polisemántico de un lenguaje tiende a incrementar la riqueza de su componente semántico, haciendo casi imposible su formalización.

Podemos resumir que los lenguajes naturales se distinguen por las siguientes propiedades:

- Han sido desarrollados por enriquecimiento progresivo antes de cualquier intento de formación de una teoría.
- La importancia de su carácter expresivo es debida fundamentalmente a la riqueza del componente semántico (polisemia).
- Existe dificultad o imposibilidad de una formalización completa.

3.2.2 Lenguaje formal

Un lenguaje formal es un lenguaje artificial compuesto por símbolos y fórmulas, que tiene como objetivo fundamental formalizar la programación de computadoras o representar simbólicamente el conocimiento científico.

Las palabras y oraciones en un lenguaje formal están perfectamente definidas, en donde una palabra mantiene el mismo significado prescindiendo del contexto o su uso.

Los lenguajes formales son exentos de cualquier componente semántico fuera de sus operadores y relaciones, y es gracias a esta ausencia de significado especial que los lenguajes formales pueden ser usados para modelar una teoría de la mecánica, de la ingeniería eléctrica, en la lingüística u otra naturaleza. Esto equivale a decir que durante la concepción de lenguajes formales toda la ambigüedad es eliminada.

En resumen, las características de los lenguajes formales son las siguientes:

- Se desarrollan de una teoría preestablecida.
- Tiene componente semántico mínimo.
- Posibilidad de incrementar el componente semántico de acuerdo con la teoría a formalizar.
- La sintaxis produce oraciones no ambiguas.
- Los números tienen un rol importante.
- Poseen una completa formalización y por esto, potencialmente posibilitan la construcción computacional.

3.3 Procesamiento del lenguaje natural

El estudio del lenguaje está relacionado con varias disciplinas. Una de ellas es la lingüística general, que estudia la estructura general y descubre las leyes universales de funcionalidad de los lenguajes naturales. Estas estructuras y leyes, aunadas a los métodos computacionales forman la lingüística computacional.

La lingüística computacional puede ser considerada como un sinónimo de procesamiento de lenguaje natural, ya que su tarea principal es la construcción de programas que procesen palabras y textos en lenguaje natural (Bolshakov & Gelbukh 2004; Sidorov 2005a).

Para llevar a cabo esta tarea, los sistemas de procesamiento de lenguaje natural deben tener conocimiento acerca de la estructura del lenguaje, para así poder pasar de texto a significado y viceversa. Los niveles que componen esta estructura se explican en el siguiente punto.

3.4 Niveles del lenguaje

La lingüística general comprende 5 niveles principales para el análisis de la estructura del lenguaje (Bolshakov & Gelbukh 2004) que son:

a) Nivel fonológico: trata de los sonidos que componen el habla, permitiendo formar y distinguir palabras.

b) Nivel morfológico: trata sobre la estructura de las palabras y las leyes para formar nuevas palabras a partir de unidades de significado más pequeñas llamadas morfemas.

c) Nivel sintáctico: trata sobre cómo las palabras pueden unirse para construir oraciones y cuál es la función que cada palabra realiza en esa oración.

d) Nivel semántico: trata del significado de las palabras y de cómo se unen para dar significado a una oración.

e) Nivel pragmático: estudia la intención del hablante al producir oraciones específicas o textos en una situación específica.

3.5 Ambigüedad

La ambigüedad, en el proceso lingüístico, se presenta cuando pueden admitirse distintas interpretaciones a partir de una representación dada o cuando existe confusión al tener diversas estructuras y no tener los elementos necesarios para eliminar las eventualmente incorrectas. Para desambiguar, es decir, para seleccionar los significados o las estructuras más adecuadas de un conjunto conocido de posibilidades, se requieren diversas estrategias de solución según el caso (Galicia-Haro 2000).

Debido a que existe ambigüedad aún para los humanos, su solución no es sólo lograr la asignación del sentido único por palabra en el análisis de textos, sino eliminar la gran cantidad de variantes que normalmente existen. La ambigüedad es el problema más importante en el procesamiento de textos en lenguaje natural, por lo que su resolución es la tarea más importante a llevar a cabo.

3.5.1 Tipos de ambigüedad

Se distinguen tres tipos principales de ambigüedad: léxica, sintáctica (estructural) y semántica.

La ambigüedad léxica se presenta cuando las palabras pueden pertenecer a diferentes categorías gramaticales, por ejemplo *bajo* puede ser una preposición, un sustantivo, un adjetivo o una conjugación del verbo *bajar* (Sidorov 2005b).

La ambigüedad sintáctica, también conocida como ambigüedad estructural se presenta cuando una oración puede tener más de una estructura sintáctica. Por ejemplo de la oración "*María habló con el profesor del instituto*" se puede entender dos cosas diferentes: a) el profesor pertenece al instituto, o bien, b) el tema del que habló María con el profesor fue el instituto.

A continuación se explica detalladamente la ambigüedad semántica y cómo resolverla.

3.6 Ambigüedad semántica

La ambigüedad semántica se presenta cuando las palabras tienen múltiples significados, por ejemplo la palabra *banco* puede significar institución financiera, la orilla del lago, asiento, etc.

Hoy en día, cualquier palabra que usamos para comunicarnos tiene dos o más posibles interpretaciones, llamadas sentidos. Para entender correctamente un texto, el lector –humano o programa de computadora– debe ser capaz de determinar el sentido adecuado para cada palabra en el texto.

Además de entender un texto, hay muchas aplicaciones de procesamiento de lenguaje natural donde la determinación automática del sentido correcto de una palabra es crucial. Entre ellas se encuentran:

a) ***Traducción automática***. La desambiguación semántica es esencial para la traducción apropiada de palabras como *bank* (*banco)* que, dependiendo del contexto, puede traducirse como *institución bancaria*, *orilla del río*, etc. (Weaver 1949) y (Yngve 1955).

b) ***Recuperación de información***. Al realizar búsquedas por palabras claves específicas, es necesario eliminar los documentos donde se usa la palabra o palabras en un sentido diferente al deseado; por ejemplo, al buscar referencias sobre animales con la palabra gato, es deseable eliminar los documentos que contienen dicha palabra asociada con mecánica automotriz (Salton 1968), (Salton & McGill 1983), (Krovetz & Croft 1992), (Voorhees 1993), (Schütze & Pedersen 1995).

c) ***El procesamiento de texto.*** La desambiguación es necesaria para algunas tareas de procesamiento de texto, por ejemplo, para determinar cuándo deben insertarse

acentos diacríticos (Yarowsky 1994) y para la detección y corrección del malapropismo (Hirst1998).

d) Etc.

3.7 Desambiguación del sentido de las palabras (WSD)

En general, la desambiguación del sentido de las palabras es el problema de seleccionar un sentido de un conjunto de posibilidades predefinidas para una palabra dada en un texto o discurso.

En los últimos años se han incrementado las investigaciones para crear métodos de WSD. A continuación se describe la clasificación para métodos de WSD de acuerdo a los recursos que utilizan (Figura 1).

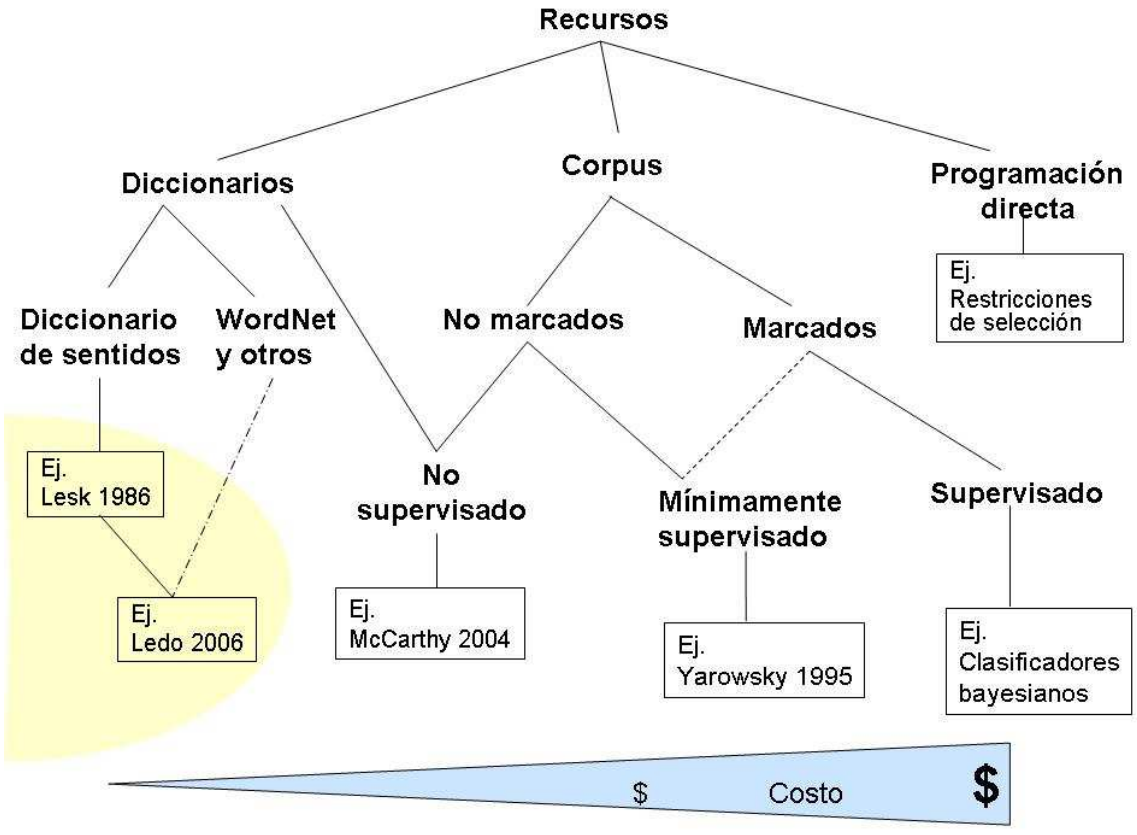

Figura 1. Clasificación de los métodos para WSD de acuerdo a los recursos que utilizan.

3.7.1 Métodos para WSD

Los métodos para desambiguación de sentidos de palabras se clasifican en: los que utilizan diccionarios, los que utilizan corpus, y los que no utilizan ningún recurso léxico.

- Los que utilizan ***diccionarios***:

Los diccionarios pueden ser **de sentidos** y otros como **WordNet.**

Los diccionarios proporcionan una lista de glosas (definición de sentido) para las palabras. Los métodos que utilizan sólo **diccionarios de sentidos**, buscan elegir un sentido (de esta lista) para cada palabra en un texto dado, tomando en cuenta el contexto en el que aparece. Como ejemplo, Lesk (1986) propone utilizar la coherencia global del texto, es decir, el total de sentidos de palabras relacionadas en el texto: mientras más relacionadas estén las palabras entre si, más coherente será el texto.

Además existen variantes del algoritmo de Lesk que utilizan no sólo **diccionarios de sentidos**, sino también otro tipo de diccionarios como **WordNet**.

- Los que utilizan ***corpus***:

Los corpus pueden ser **no marcados** y **marcados**.

Los métodos que utilizan corpus **no marcados** son los *no supervisados*, estos métodos también utilizan otros recursos como **WordNet** para poder asignar un sentido a cada palabra que aparece en los textos no marcados. Como ejemplo de éstos tenemos el método de McCarthy *et al*. (2004), el cual elige de un diccionario (tesauro) las palabras relacionadas con la palabra a desambiguar. Cada palabra relacionada tiene un peso, éstas y la palabra a desambiguar tienen sentidos en un diccionario. Para elegir el sentido correcto, las palabras relacionadas votan por un sentido de la palabra a desambiguar con cierto peso. Se elige el sentido con más peso.

Los métodos que utilizan corpus **marcados** son los métodos *supervisados*. Éstos reducen la desambiguación de sentidos de palabras a un problema de clasificación, donde

a una palabra dada se le asigna el sentido más apropiado de acuerdo a un conjunto de posibilidades, basadas en el contexto en el que ocurre. Hay muchos algoritmos de aprendizaje supervisado utilizados para WSD, como ejemplo tenemos los clasificadores bayesianos, maquinas de soporte vectorial, árboles y listas de decisión, etc.

Hay métodos que utilizan una gran cantidad de corpus **no marcados** y muy pocos **marcados** llamados *mínimamente supervisados*. Como ejemplo de éstos tenemos el método de Yarowsky (1995), el cual identifica todas las ocurrencias de una palabra a desambiguar en un corpus no marcado. Después identifica un número pequeño de colocaciones semilla representativos de cada sentido de la palabra y etiqueta todos los ejemplos que contienen la colocación semilla con la palabra de dicha colocación (así tenemos los conjuntos etiquetados con cada sentido representativo y el conjunto residuo). El algoritmo utiliza los conjuntos etiquetados para entrenar una lista de decisión y encontrar nuevas colocaciones, para después etiquetar sobre el conjunto residuo. El algoritmo termina cuando el conjunto residuo se estabiliza.

- Los que utilizan **programación directa**:

Estos métodos se basan en reglas (muchas) que especifican el sentido de una palabra de acuerdo al contexto en el que aparece. Un ejemplo son las restricciones de selección (*selectional restrictions*), las cuales definen reglas de acuerdo a la palabra a desambiguar y su argumento. Ejemplo: el verbo comer puede tener como restricción que su tema argumento sea comida (comer-comida).

Este trabajo se enfoca en los métodos que se basan en la aplicación directa de diccionarios de sentidos, que se describen a continuación.

3.7.2 *Back-off* para métodos de WSD

Los métodos de *back-off* o de respaldo se utilizan como una herramienta, que aunada al método principal aumenta la cobertura de casos en los que decide el sistema, es decir, existen casos en los que el método principal no tiene suficiente información para elegir el

sentido de una palabra, de tal forma que el método de *back-off* toma la decisión en estos casos.

Los principales de métodos de *back-off* utilizados son sentido aleatorio y sentido más frecuente.

3.7.2.1 Método de *back-off* a sentido aleatorio

Este método consiste en elegir un sentido de forma aleatoria para las palabras en las que el método principal no tomó ninguna decisión. El sentido aleatorio es elegido de la lista de sentidos dada por un diccionario para las palabras.

La cobertura del método principal aunado a este tipo de *back-off* es, en la mayoría de los casos, de 100%; ya que existen palabras que no están definidas en el diccionario y no es posible elegir un sentido para ellas.

3.7.2.2 Método de *back-off* a sentido más frecuente

Este método consiste en elegir el sentido más frecuente para las palabras en las que el método principal no tomó ninguna decisión. El sentido más frecuente de una palabra está dado por la frecuencia con la que éste aparece en un corpus determinado. En el caso del diccionario WordNet el sentido más frecuente de una palabra es el primer sentido definido.

La cobertura del método principal aunado a este tipo de *back-off* es, en la mayoría de los casos, de 100%; ya que existen palabras que no están definidas en el diccionario y no es posible elegir un sentido para ellas.

3.8 Algoritmos tipo Lesk

Para poder entender el algoritmo de Lesk así como el de Lesk simplificado, es necesario entender los términos siguientes:

a) Ventana de contexto: En desambiguación del sentido de las palabras, ventana de contexto se refiere a las palabras vecinas a la palabra ambigua. Dicha ventana es de tamaño variable, es decir, desde dos palabras, oración, párrafo, texto, etc.

b) Coherencia global: Al hablar de desambiguación del sentido de las palabras, la coherencia global se refiere a que todos los sentidos de todas las palabras en cierta *ventana de contexto*, están co-relacionados, de forma que el sentido de una palabra está coordinado con el sentido de las demás. Es decir, la coherencia global es el hecho de decidir simultáneamente el sentido para todas las palabras en cierta *ventana de contexto*.

c) Coherencia local: Al hablar de desambiguación del sentido de las palabras, la coherencia local se refiere a que el sentido de una palabra no está coordinado con los sentidos de las demás palabras de su *ventana de contexto.* Es decir, la coherencia local es el hecho de decidir el sentido de una palabra a la vez sin tomar en cuenta el sentido de las demás palabras.

3.8.1 Algoritmo de Lesk

El algoritmo de Lesk (1986) es uno de los primeros algoritmos exitosos usados en la desambiguación de sentidos de palabras. Este algoritmo está determinado por dos principales ideas: un algoritmo de optimización para WSD y una medida de similitud para las definiciones de los sentidos.

El primero es acerca de desambiguar palabras considerando la coherencia global del texto, esto es, encontrar la combinación de los sentidos que maximice la relación total entre los sentidos de todas las palabras.

Por ejemplo, para la oración *My father deposits his money in a bank account* y considerando a lo más tres sentidos[1] (véase tabla 3), para cada palabra, la figura 2 muestra la representación del algoritmo de Lesk original.

[1] Sentidos obtenidos de WordNet

Tabla 3. Sentidos de las palabras (máximo tres) obtenidas de WordNet para la oración "*My father deposits his money in a bank account*".

My **father deposits** his **money** in a **bank account**
Father 1: a male parent (also used as a term of address to your father); "his father was born in Atlanta". 2: `Father' is a term of address for priests in some churches (especially the Roman Catholic Church or the Orthodox Catholic Church); "`Padre' is frequently used in the military". 3: a person who holds an important or distinguished position in some organization; "the tennis fathers ruled in her favor"; "the city fathers endorsed the proposal".
Deposit 1: fix, force, or implant; "lodge a bullet in the table". 2: put into a bank account; "She deposits her paycheck every month". 3: put (something somewhere) firmly; "She posited her hand on his shoulder"; "deposit the suitcase on the bench"; "fix your eyes on this spot".
Money 1: the official currency issued by a government or national bank; "he changed his money into francs".
bank 1: a financial institution that accepts deposits and channels the money into lending activities; "he cashed a check at the bank"; "that bank holds the mortgage on my home". 2: sloping land (especially the slope beside a body of water); "they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents". 3: a supply or stock held in reserve for future use (especially in emergencies)
account 1: a formal contractual relationship established to provide for regular banking or brokerage or business services; "he asked to see the executive who handled his account". 2: the act of informing by verbal report; "he heard reports that they were causing trouble"; "by all accounts they were a happy couple". 3: a record or narrative description of past events; "a history of France"; "he gave an inaccurate account of the plot to kill the president"; "the story of exposure to lead".

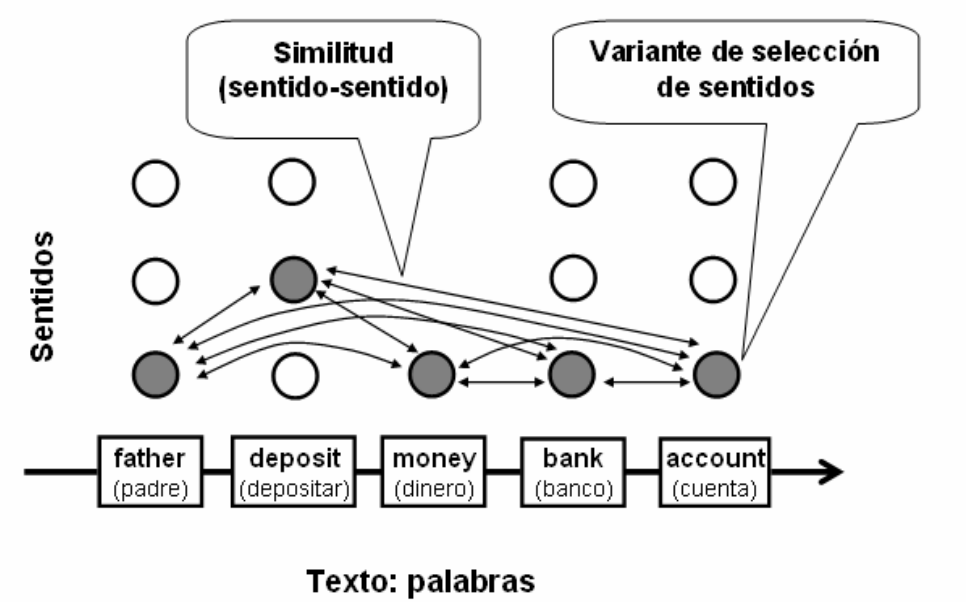

Figura 2. Representación gráfica del algoritmo original de Lesk.

En el segundo punto, relacionado con la medida de similitud, Lesk sugiere usar el traslape entre las definiciones de los sentidos, es decir, contar el número de palabras que tienen en común.

Como ejemplo, para la oración, "*My father deposits his money in the bank account*" para medir la relación de las definiciones de los sentidos para la palabra *"deposit"* y "*bank"* como Lesk lo propuso, es necesario contar las palabras en común en todas las definiciones. En este caso, comparando principalmente las tres definiciones de *"deposit"* contra las tres definiciones de *"bank"*. La relación entre los valores se muestra en la siguiente tabla.

Tabla 4. Valores de relación para las definiciones de sentidos de las palabras "deposit" y "bank".

Número de sentidos para *deposit*	Número de sentidos para *bank*	Valor de relación
1	1	0
1	2	0
1	3	0
2	**1**	**2**
2	2	1
2	3	0
3	1	1
3	2	0
3	3	0

La figura 3 muestra los pasos principales para el método original de Lesk. En el pseudocódigo esta resaltado en itálica y negrita donde la función corresponde a la medida de similitud propuesta por Lesk. Todo el pseudocódigo mostrado se refiere al algoritmo de optimización para WSD.

```
for each vector v = (s1, ... sn) where si in vi
    sumv = 0;
    for each i
            for each j
                    sumv += relatedness-overlap(si, sj)
Chose the vector v with the biggest value of sumv
for each word w
    sense(w) = v[w]
```

Figura 3. Pseudocódigo del método de Lesk original

Por un lado, la limitación principal de la medida de similitud propuesta por Lesk, es que las glosas del diccionario, regularmente, son muy cortas y no incluyen el vocabulario suficiente para identificar los sentidos relacionados (Patwardhan *et al.* 2003). Esta es la razón de porqué diferentes autores han propuesto distintas medidas de similitud (Resnik 1995; Jiang & Conrath 1997; Hirst & St-Onge 1998; Leacock & Chodorow 1998; Lin 1998) (véase sección 3.8.3).

Como se mencionó con anterioridad, el algoritmo original de Lesk es muy caro computacionalmente hablando, debido a que mientras más palabras tenga el texto, y más sentidos por cada palabra, mayor será el número de combinaciones de sentidos, el cual se incrementa de manera exponencial, haciéndolo prácticamente prohibitivo para una búsqueda exhaustiva que garantice encontrar el óptimo global exacto. Por ejemplo, para una oración de 16 palabras de contenido, donde cada palabra contiene siete sentidos (números cercanos a los observados en el corpus de SemCor[1]), existen $7^{16} \approx 3 \times 10^{13}$ posibles combinaciones a escoger, de las cuales una será seleccionada.

Para aliviar el problema de la complejidad computacional del algoritmo original de Lesk, dos principales modificaciones al algoritmo han sido propuestas, a) una versión simplificada que considere las palabras, una por una, y compare cada sentido de la palabra dada con el contexto, y b) el uso de búsquedas basadas en heurísticas para encontrar una solución óptima cercana a la real (Véase Capítulo 4).

3.8.2 Lesk simple o Lesk simplificado

Para reducir el espacio de búsqueda del algoritmo original de Lesk, Kilgarriff y Rosenzweig (2000) propusieron una variación del algoritmo original de Lesk, conocido como algoritmo de Lesk simplificado o Lesk simple, donde los sentidos de las palabras en el texto son determinados uno a uno encontrando el mayor traslape entre los sentidos de las definiciones de cada palabra con el contexto actual, véase la figura 4. En lugar de buscar asignar, simultáneamente, el significado de todas la palabras en un texto dado, este enfoque determina el sentido de las palabras uno a uno, por lo que se evita la

explosión combinatoria de sentidos. La figura 5 muestra los principales pasos del algoritmo.

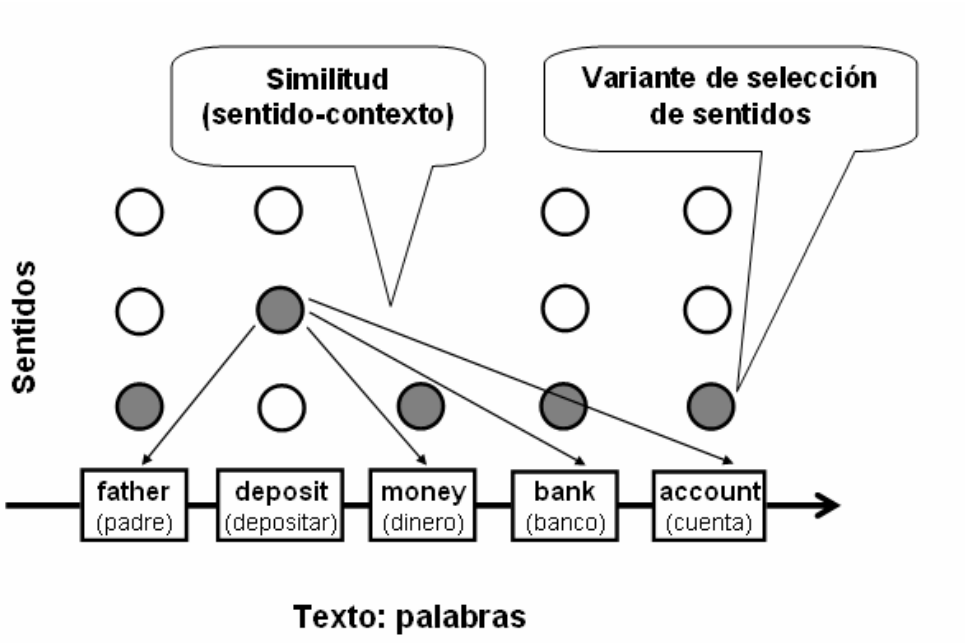

Figura 4. Representación gráfica del algoritmo de Lesk simplificado.

```
for each word w
  for each sense i of w
        x_i = overlap (s_i, context)
  chose sense i for w, where x_i is maximized
```

Figura 5. Pseudocódigo del algoritmo de Lesk simple

3.8.3 Medidas de similitud semántica

Como se mencionó en la sección 3.8, la medida de similitud propuesta por Lesk, traslape (*overlapping*), tiene la limitación del tamaño de las glosas del diccionario que, regularmente son muy cortas.

A continuación se describen ciertas medidas de similitud las cuales han sido propuestas para medir la proximidad semántica entre dos sentidos o palabras, usando WordNet como espacio semántico.

3.8.3.1 Medida de Lesk Adaptada

Lesk propuso medir la similitud entre sentidos contando el traslape de palabras. La limitación principal de esta técnica es que las glosas del diccionario, por lo general, son muy breves, de tal manera que no incluyen suficiente vocabulario para identificar los sentidos relacionados. Banerjee y Pedersen (2002), sugieren una adaptación del algoritmo basado en WordNet. Esta adaptación consiste en tomar en cuenta las glosas de los vecinos de la palabra a desambiguar, explotando los conceptos jerárquicos de WordNet, de tal manera que las glosas de los vecinos son expandidas incluyendo a su vez las glosas de las palabras con las cuales se encuentran relacionadas mediante las diversas jerarquías que presenta WordNet. Banerjee y Pedersen, sugieren una variación en la manera de asignar el puntaje a una glosa, de tal manera que si "n" palabras consecutivas son iguales en ambas glosas, estas deberán de tener mayor puntaje que aquel caso en el que sólo coincide una sola palabra en ambas glosas.

Supongamos que *bark* (ladrido o corteza) es la palabra que se desea desambiguar y sus vecinos son *dog* (perro) y *tail* (cola). El algoritmo original de Lesk checa las coincidencias en las glosas de los sentidos de *dog* con las glosas de *bark*. Luego checa las coincidencias en las glosas de *bark* y *tail*. El sentido de *bark* con el máximo número de coincidencias es seleccionado. La adaptación del algoritmo de Lesk considera estas mismas coincidencias y añade además las glosas de los sentidos de los conceptos que se encuentran relacionados semántica o léxicamente a *dog, bark* y *tail*, de acuerdo a las jerarquías de WordNet.

3.8.3.2 Medida de Leacock-Chodorow

Esta medida está basada en las longitudes de rutas usando la jerarquía "es-un" de WordNet, para las definiciones de sustantivos (Leacock *et al.* 1998). La ruta más corta entre dos conceptos es aquella que incluye el menor número de conceptos intermedios.

Este valor es escalado por la profundidad de la jerarquía, donde dicha profundidad es definida como la longitud desde el nodo raíz hasta un nodo hoja. Por consiguiente la medida de relación esta definida por la siguiente fórmula:

$$relación_{lch}(c_1, c_2) = \max[-\log(RutaMasCorta(c_1, c_2)/(2.D))]$$

$RutaMasCorta(c_1, c_2)$ es la longitud de la ruta más corta entre dos conceptos (ruta con menor número de nodos) y D es la profundidad máxima de la taxonomía (distancia entre la raíz y el nodo más alejado de ésta). La implementación de esta medida usando WordNet, asume un nodo raíz hipotético que junta todas las jerarquías de sustantivos, de tal manera que D llega a ser una constante de 16 para todos los sustantivos, lo cual significa que la longitud entre el nodo raíz y la hoja más lejana del árbol es de 16.

3.8.3.3 Medida de Resnik

Resnik (1995) introduce una medida de relación basada en el concepto de "contenido de la información" más conocido en inglés como *information content*, el cual se trasluce como un valor que es asignado a cada concepto en una jerarquía basada en la evidencia encontrada en un corpus.

El término "contenido de la información" es una simple medida de la especificación de un concepto. Un concepto con un gran contenido de información es muy específico a un tópico particular, mientras que conceptos con un contenido de información bajo están asociados con tópicos más generales. Por lo tanto, la expresión *carving fork* (tenedor) tiene un alto contenido de información, mientras que *entity* (entidad) tiene un bajo contenido de información.

El contenido de información de un contexto es estimado contando la frecuencia de ese concepto en un corpus de gran escala, determinando de esta manera su probabilidad. De acuerdo a Resnik, el logaritmo negativo de esta probabilidad determina el contenido de información del concepto.

$$IC(concept) = -\log(P(concept))$$

Si se tuviera un texto etiquetado de sentidos, contar la frecuencia de un concepto sería logrado directamente, ya que cada concepto sería asociado con un único sentido; pero en caso contrario, Resnik sugiere contar el número de ocurrencias de una palabra en el corpus y luego dividir dicho valor por el número de sentidos que tiene dicho término, siendo este valor asignado a cada concepto. Por ejemplo, supongamos que la palabra *bank* (banco) ocurre 20 veces en un corpus, y existen dos conceptos asociados a dicha palabra en una jerarquía, uno para *river bank* (orilla de río) y el otro para *financial bank* (institución financiera). Cada uno de estos conceptos recibirá un valor de 10; en cambio si las ocurrencias de *bank*, se presentaran en un texto etiquetado con sentidos, la información sería más consistente.

La frecuencia de un concepto incluye la frecuencia de todos sus conceptos subordinados, ya que el conteo de un concepto es añadido a su inmediato superior. Es necesario notar que los conteos de los conceptos más específicos son añadidos a los más genéricos; y no de manera contraria; por ende los conteos de los conceptos específicos incrementan el total de los más genéricos. Dichos conceptos tendrán una mayor probabilidad asociada, lo que significaría que tendrían un bajo "contenido de información", ya que estos representan conceptos muy generales.

La medida de Resnik usa el "contenido de información" de conceptos dentro de las jerarquías "es-un". La idea principal detrás de esta medida es que dos conceptos están semánticamente relacionados teniendo en cuenta la cantidad de información que ellos comparten en común. La cantidad de información común de dos conceptos es determinada por el "contenido de información" del concepto más bajo (*lowest common subsumer*) para las dos conceptos en cuestión. La medida de Resnik es calculada con la siguiente fórmula:

$$sim_{res}(c_1, c_2) = IC(lcs(c_1, c_2))$$

Esta medida no considera el contenido de información del par de conceptos a comparar y tampoco considera la longitud de la ruta entres ambos. La principal limitante de esta técnica es que algunos pares de conceptos compartirían el mismo valor de similitud, ya que existe la posibilidad de que el mismo *lowest common subsumer* sea asignado a más de un par de conceptos. Por ejemplo, *vehicle* (vehículo) es el *lowest*

common subsumer de *jumbo jet* (avión jumbo), *tank* (tanque) y *house trailer* (remolque). Por consiguiente estas parejas recibirían el mismo puntaje en su comparación.

3.8.3.4 Medida de Jiang-Conrath

Jiang y Conrath (1997) usan el concepto de "contenido de información" planteado por Resnik, al cual lo complementan con las longitudes de rutas entre conceptos. Esto resulta una técnica híbrida para computar la relación semántica de una pareja de conceptos. Esta técnica incluye el "contenido de información" de los propios conceptos y del *lowest common subsumer*. Esta medida es determinada por la siguiente fórmula:

$$dist_{jcn}(c_1, c_2) = IC(c_1) + IC(c_2) - 2 \times IC(lcs(c_1, c_2))$$

3.8.3.5 Medida de Lin

La medida de Lin (1997) está basada en su teorema de similitud. Este establece que la similitud de dos conceptos es medida por la razón entre la cantidad de información necesaria para establecer la información común de ambos conceptos y la cantidad de información necesaria para describirlos. Esta información común entre dos conceptos es obtenida por el contenido de información del *lowest common subsumer* que aplica para ambos conceptos y el contenido de información de cada uno los conceptos propiamente dichos.

Esta medida es muy parecida a la presentada por Jiang y Conrath; aunque ellas fueros desarrolladas independientemente. Esta medida es determinada por la siguiente fórmula:

$$related_{lin}(c_1, c_2) = \frac{2 \times IC(lcs(c_1, c_2))}{IC(c_1) + IC(c_2)}$$

Esta medida puede ser vista como la intersección del contenido de información de los dos conceptos a comparar dividido por la suma del contenido de información de ambos.

3.8.3.6 Medida vector

Esta medida, al igual que la de Lesk, incorpora información de las glosas de WordNet. La medida *vector*, crea una matriz de co-ocurrencia para cada palabra usada en las glosas de WordNet tomando cualquier corpus y luego representa cada glosa con un vector que es el promedio de los vectores de co-ocurrencia.

3.8.3.7 Medida path

Este medida calcula la relación semántica de sentidos contando el número de nodos junto con el camino mas corto entre el sentido en la jerarquía "es-un" de WordNet. La longitud de los caminos incluye los nodos terminales.

Dado que un camino largo indica menos relación, el valor de relación obtenido es el inverso multiplicativo de la longitud del camino (distancia) entre los dos conceptos: relación = 1/distancia. Si los dos conceptos son idénticos, entonces la distancia entre ellos es uno; por lo tanto, su relación también es 1. Sí no es encontrado ningún camino, entonces un valor negativo muy grande es devuelto.

3.8.3.8 Medida combinada

En realidad no es una medida de similitud, sino la combinación de tres de ellas en base a la categoría gramatical de las palabras.

Sinha y Mihalcea (2007) hicieron un estudio de las medidas de similitud que daban mejores resultados para cada par de categorías gramaticales. JCN funciona mejor cuando el par de palabras para obtener su relación de similitud son sustantivos. LCH funciona mejor para verbo, verbo. Para todo lo demás funciona mejor la medida de LESK.

3.9 Evaluación de sistemas de WSD

En la actualidad existen muchos métodos para la desambiguación del sentido de las palabras. Al principio, cada uno de ellos aplicaba su propio esquema de evaluación, lo que hacia una tarea complicada el llevar a cabo comparaciones entre los distintos

métodos (Agirre *et al.* 2007). Recientemente, se ha utilizado un nuevo enfoque para la evaluación de los sistemas de desambiguación de sentidos de las palabras, conocida como basados en corpus o en inglés *corpus-based*. Dicho enfoque unifica los sistemas de evaluación, lo que permite llevar a cabo comparaciones entre diferentes algoritmos.

Existen dos tipos de evaluaciones para los sistemas de desambiguación de sentidos de las palabras (Ide & Verónis 1998). En la primera, conocida como *in Vitro*, la tarea de WSD esta definida independientemente de cualquier aplicación, y los sistemas son comparados usando *benchmarks* (sistemas de referencia) especializados conocidos como corpus. La segunda, conocida como *in Vivo*, asigna una puntuación en términos de la contribución que tiene el sistema, al desempeño de una aplicación especifica en el procesamiento de lenguaje natural.

El tipo de evaluación *in Vitro* es uno de los más utilizados. Para este tipo de evaluación es necesario contar con: un corpus etiquetado manualmente, un diccionario y un sistema de puntuación que asigna un cierto valor para los aciertos y los errores. Este sistema de evaluación es el utilizado en esta tesis.

A continuación se describen los diccionarios y corpus más utilizados en la tarea de desambiguación del sentido de las palabras y por último las medidas de evaluación utilizadas.

3.9.1 Diccionario

El uso de un diccionario en la desambiguación del sentido de las palabras es de suma importancia, ya que ésta consiste en asignar, a cada palabra en un texto dado, un sentido que se relaciona con una lista de sentidos en un diccionario. En el método propuesto es utilizado WordNet 2.1 como diccionario para obtener los sentidos de una palabra; y además se utiliza para la obtención de métricas de similitud semántica (sección 3.8.3).

3.9.1.1 WordNet

WordNet es un sistema de referencia léxico, el cual fue desarrollado en la universidad de Princeton bajo la dirección del profesor George A. Miller. Este recurso combina muchas características usadas para desambiguación del sentido de las palabras en un solo sistema.

WordNet, incluye definiciones de sentidos de palabras como un diccionario, define synsets o conjuntos de sinónimos, los cuales representan un concepto léxico, y además proporciona las relaciones existentes entre palabras.

WordNet 2.1 cuenta con 155327 palabras, 117597 *synsets* y 207016 sentidos; está conformado por tres bases de datos correspondientes a sustantivos, verbos y una para adjetivos y adverbios. Cada base de datos está conformada por entradas léxicas que corresponden a formas ortográficas individuales. Cada palabra es asociada con un conjunto de sentidos. La tabla 5 muestra un ejemplo basado en el sustantivo *person* (persona).

Tabla 5. Sentidos dados por WordNet 2.1 para el sustantivo *person* (persona).

Sentido	Definición de glosa
1	(7229) person, individual, someone, somebody, mortal, soul – (a human being, "there was too much for one person to do")
2	(11) person – (a human body (usually including the clothing); "a weapon was hidden on his person")
3	person – (a grammatical category of pronouns and verb forms; "stop walking about yourself in the third person")

Los sentidos de WordNet comprenden un conjunto de sinónimos y definiciones al igual que un diccionario, las cuales son llamadas glosas. El número que se encuentra al inicio de la definición de algunos sentidos, es la frecuencia de los valores obtenidos del corpus SemCor. A diferencia de un diccionario, WordNet contiene un conjunto de relaciones léxicas entre *synsets* o lemas, los cuales aparecen al inicio de la glosa.

WordNet proporciona mucha más información léxica que no será explicada en este documento.

3.9.2 Corpus

Un corpus es un conjunto de datos compuesto regularmente por un gran número de oraciones, en los que a las palabras se les asigna un sentido de un diccionario dado. Entre los corpus, anotados sintáctica y semánticamente, que normalmente se usan para esta tarea, se encuentran Senseval-2, Senseval-3, Semeval y SemCor. A continuación se describen cada uno de ellos, aunque nuestros experimentos se llevaron a cabo sólo sobre

Senseval-2, debido a que las evaluaciones en el estado del arte son en su mayoría sobre este corpus.

3.9.2.1 Senseval

Senseval es una organización internacional dedicada a la evaluación de sistemas de desambiguación del sentido de las palabras (WSD).

Su misión consiste en organizar y ejecutar las actividades relacionadas con la evaluación y prueba a los puntos fuertes y débiles de los sistemas de WSD con respecto a las distintas palabras, los diferentes aspectos del idioma, y diferentes idiomas. Cada tres años se llevan a cabo estas actividades en un taller conocido como "Senseval: International Workshop on Evaluating Word Sense Disambiguation Systems". El 2007 se ha llevado a cabo Senseval 4, renombrado "Semeval: Internacional Workshop on Semantic Evaluations".

En estos talleres se evalúan muchas tareas semánticas. Una de ellas es desambiguación de sentidos para todas las palabras en inglés llamada "English all-words". En esta tarea se proporciona un corpus etiquetado sintácticamente para que los participantes apliquen sus sistemas y obtengan los sentidos, en relación con un diccionario específico, para todas las palabras. En el anexo 1 se muestra el significado de las etiquetas usadas en el corpus Senseval y algunos ejemplos.

3.9.2.1.1 Senseval 2

Los datos de Senseval 2 constan de 3 archivos con un total de 238 oraciones y 2436 palabras para desambiguar, de las cuales, 1136 son sustantivos, 544 son verbos, 457 son adjetivos y 299 son adverbios. En la tabla 6 se puede ver la distribución de estos datos (de acuerdo su categoría gramatical) para cada archivo.

Tabla 6. Datos de la categoría gramatical de las palabras de Senseval 2

	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
ARCHIVO 1	99	86	331	162
ARCHIVO 2	170	107	495	242
ARCHIVO 3	188	106	310	140

3.9.2.1.2 Senseval 3

Los datos de Senseval 3 constan de 3 archivos con un total de 300 oraciones y 2081 palabras para desambiguar, de las cuales, 951 son sustantivos, 751 son verbos, 364 son adjetivos y 15 son adverbios. En la tabla 7 se puede ver la distribución de estos datos (de acuerdo su categoría gramatical) para cada archivo.

Tabla 7. Datos de la categoría gramatical de las palabras de Senseval 3

	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
ARCHIVO 1	103	11	327	364
ARCHIVO 2	174	1	339	153
ARCHIVO 3	87	3	285	234

3.9.2.1.3 Semeval

Los datos de Semeval constan de 3 archivos con un total de 126 oraciones y 490 palabras para desambiguar, de las cuales, 163 son sustantivos y 327 son verbos. En la tabla 8 se puede ver la distribución de estos datos (de acuerdo su categoría gramatical) para cada archivo.

Tabla 8. Datos de la categoría gramatical de las palabras de Semeval

	SUSTANTIVOS	VERBOS
ARCHIVO 1	44	76
ARCHIVO 2	60	102
ARCHIVO 3	59	149

3.9.2.2 SemCor

SemCor, creado por la Universidad de Princeton, es un corpus etiquetado sintáctica y semánticamente. Semcor 2.1 consta de un total de 352 archivos. En 186 de ellos, sustantivos, adjetivos, verbos y adverbios son etiquetados con su categoría gramatical, lema y sentido; mientras que en los 166 textos restantes, sólo los verbos son etiquetados con su lema y sentido. El número total de *tokens* (términos) en SemCor es de 359,732 en el primer conjunto de archivos, de los cuales 192,639 han sido etiquetados semánticamente, mientras que en el segundo grupo este número asciende a 316,814 *tokens*, de los cuales 41,497 ocurrencias de verbos son etiquetados semánticamente.

En la figura 6 podemos ver un ejemplo del formato y etiquetado que usa SemCor para representar oraciones ("*The City_Purchasing_Department the jury said is lacking in experienced clerical personnel as a result of city personnel policies*). La información de cada palabra en una oración se encuentra delimitada por las etiquetas <wf > y </wf>.

```
<p pnum=9>
<s snum=9>
<wf cmd=ignore pos=DT>The</wf>
<wf cmd=done rdf=group pos=NNP lemma=group wnsn=1 lexsn=1:03:00::
pn=group>City_Purchasing_Department</wf>
<punc>,</punc>
<wf cmd=ignore pos=DT>the</wf>
<wf cmd=done pos=NN lemma=jury wnsn=1 lexsn=1:14:00::>jury</wf>
<wf cmd=done pos=VB lemma=say wnsn=2 lexsn=2:32:01::>said</wf>
<punc>,</punc>
<punc>``</punc>
<wf cmd=done pos=VB lemma=be wnsn=1 lexsn=2:42:03::>is</wf>
<wf cmd=done pos=JJ lemma=lacking wnsn=2 lexsn=5:00:00:inadequate:00>lacking</wf>
<wf cmd=ignore pos=IN>in</wf>
<wf cmd=done pos=JJ lemma=experienced wnsn=1 lexsn=3:00:00::>experienced</wf>
<wf cmd=done pos=JJ lemma=clerical wnsn=1 lexsn=3:01:01::>clerical</wf>
<wf cmd=done pos=NN lemma=personnel wnsn=1 lexsn=1:14:00::>personnel</wf>
<wf cmd=ignore pos=IN>as</wf>
<wf cmd=ignore pos=DT>a</wf>
<wf cmd=done pos=NN lemma=result wnsn=1 lexsn=1:19:00::>result</wf>
<wf cmd=ignore pos=IN>of</wf>
<wf cmd=done pos=NN lemma=city wnsn=2 lexsn=1:15:01::>city</wf>
<wf cmd=done pos=NN lemma=personnel wnsn=1 lexsn=1:14:00::>personnel</wf>
<wf cmd=done pos=NN lemma=policy wnsn=2 lexsn=1:09:00::>policies</wf>
<punc>"</punc>
<punc>.</punc>
</s>
```

Figura 6. Formato y etiquetado usado por Semcor (Oracion: *The City_Purchasing_Department, the jury said, "is lacking in experienced clerical personnel as a result of city personnel policies".*).

3.9.3 Medidas de evaluación

Para medir el desempeño de un sistema de desambiguación de sentidos de palabras, se utilizan medidas de evaluación que son: *Coverage* (cobertura), *Precision* (precisión) y *Recall*.

Coverage determina el porcentaje de casos cubiertos por el sistema. Esta dado por el número de veces que el sistema asignó un sentido entre el total de casos.

Precision es usado para medir la exactitud o fidelidad del algoritmo. En el caso de desambiguación del sentido de las palabras esta definido por, el total de sentidos correctos entre el total de casos cubiertos por el sistema.

Recall está definida por el total de sentidos correctos sobre el total de casos.

3.10 Conclusiones

En este capítulo vimos que la ambigüedad se presenta en todos los niveles del lenguaje. Uno de los problemas principales de los sistemas de procesamiento de lenguaje natural es resolver la ambigüedad.

Los tres tipos principales de ambigüedad son léxica, sintáctica y semántica. La ambigüedad semántica se presenta cuando una palabra tiene múltiples significados. Este tipo de ambigüedad es uno de los problemas más difíciles de resolver en sistemas de procesamiento de lenguaje natural.

Se presentaron los métodos para desambiguación del sentido de las palabras, más detalladamente, los métodos que se basan en la aplicación directa del diccionario de sentidos como lo son: el algoritmo de Lesk y sus variantes, entre ellas, Lesk simplificado.

Por último se explica cómo se evalúan métodos para desambiguación del sentido de las palabras; se describen los recursos lingüísticos más usados para evaluación: diccionarios, corpus y medidas de evaluación.

CAPÍTULO

4

Métodos de optimización global

Capítulo 4. Métodos de optimización global

4.1 Introducción

Una de las formas de aliviar la complejidad computacional que presenta el algoritmo original de Lesk es el uso de métodos de optimización global para encontrar la configuración más cercana al óptimo.

En este capítulo veremos qué es la optimización, tanto local como global. Daremos una clasificación de los métodos de optimización basados en heurísticas tradicionales y heurísticas modernas. Daremos una breve explicación de los métodos heurísticos modernos o metaheurísticos, ya que esta tesis se basa en algunos de ellos.

Además, explicaremos a detalle algunos de los métodos heurísticos modernos, que se han utilizado para desambiguación del sentido de las palabras, como:

- Templado simulado (Simulated Annealing)
- Algoritmos genéticos

Al final del capítulo explicaremos más detalladamente los Algoritmos con Estimación de Distribuciones (EDA), ya que han sido utilizados en esta tesis para la desambiguación del sentido de las palabras.

4.2 Optimización

De forma genérica, puede definirse la optimización como el proceso de determinar las mejores soluciones a problemas matemáticos que a menudo modelan una realidad. Los problemas complejos de optimización pueden encontrarse en todos los

campos de la ciencia. En este aspecto, la optimización numérica ha adquirido mucha atención entre la comunidad científica durante las últimas décadas, y quizás lo más confuso reside en decidir qué algoritmo de optimización se ajusta mejor a las características del problema bajo análisis.

El objetivo que se persigue al resolver un problema de optimización es encontrar una solución óptima con un coste computacional razonable. Aunando estas dos premisas puede establecerse una clasificación preliminar de los métodos de optimización en dos grandes bloques, distinguiendo por un lado los métodos de búsqueda local y, por otro, las así denominadas técnicas de optimización global.

Los métodos locales obtienen la mejor solución posible en las inmediaciones del punto inicial, atribuyéndoseles una fuerte dependencia del punto de arranque del algoritmo. La mayor parte de los métodos locales utilizan la información del gradiente, requieren el cálculo de derivadas y, en definitiva, imponen sobre el espacio de búsqueda unas condiciones de diferencia y continuidad difíciles de garantizar y controlar en la práctica en gran parte de los problemas.

Por otra parte, las técnicas de optimización global exhiben una gran independencia de la naturaleza del espacio de soluciones y, a diferencia de las técnicas de búsqueda local, son capaces de atravesar un espacio de búsqueda con múltiples mínimos o máximos locales y alcanzar una solución global al problema, entendiendo como tal la mejor solución posible o una solución en las inmediaciones de la región que contiene a la solución óptima.

Independientemente de la distinción entre técnicas locales y globales de optimización, en la Figura 7 se muestran los métodos de optimización más representativos clasificados en base a métodos heurísticos tradicionales y heurísticos modernos (metaheurísticos). Estos últimos han adquirido durante la última década una notable aceptación en diferentes campos de la ciencia, debido a que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los heurísticos tradicionales no son efectivos (Glover, 1986); y entorno a algunos de ellos se centra esta investigación.

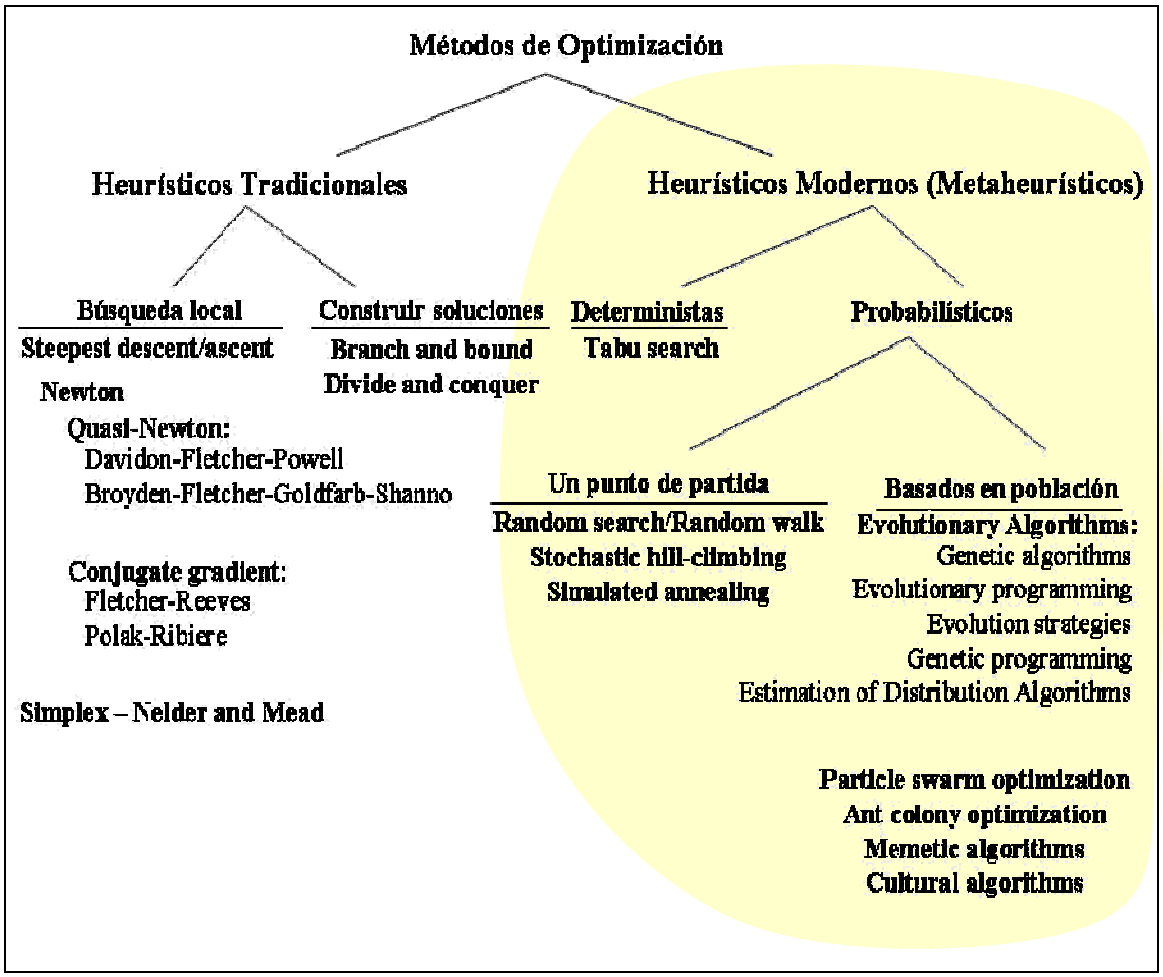

Figura 7. Clasificación de los métodos de optimización más relevantes
(por uniformidad con la literatura científica se conserva la denominación original en inglés de cada método)

La búsqueda Tabú (*Tabu search*)(Glover 1986), de naturaleza determinista, tiene capacidad para escapar de los mínimos o máximos locales, aprovechando un cierto conocimiento acerca del dominio de búsqueda y actualizando la solución en curso con el mejor punto de su vecindad.

Por otra parte, dentro de los métodos heurísticos de naturaleza probabilística, (denominación asociada con el hecho de que la optimización depende de eventos aleatorios) existen dos familias: aquellos que utilizan un único punto de partida y aquellos que utilizan una población. Todas las variantes de los métodos heurísticos probabilísticos que utilizan un único punto de partida, a excepción del templado simulado (*Simulated Annealing*), tienen unos fundamentos muy sencillos, que se limitan a hacer evolucionar una solución inicial perturbando aleatoriamente los parámetros a optimizar.

En lo que respecta al templado simulado, este método imita a nivel computacional el proceso físico a seguir para obtener sólidos con configuraciones de energía mínima (Kirkpatrick *et al*. 1983).

En línea con el templado simulado y en un intento por imitar procesos naturales como la evolución de las especies o los propios comportamientos sociales y culturales de diferentes colectivos, entre los cuales puede incluirse a los propios seres humanos, surgen nuevos métodos que establecen una nueva concepción de la optimización. Todos estos algoritmos tienen en común el hecho de utilizar una población o conjunto de soluciones potenciales y someterlos a un proceso iterativo, utilizando diferentes esquemas, operadores y estrategias en función del tipo de algoritmo. La familia más extensa de este tipo de algoritmos es la que agrupa a los así denominados algoritmos evolutivos. Estos algoritmos engloban una serie de técnicas inspiradas en los principios de la teoría de la evolución natural de Darwin. En términos generales, estos algoritmos hacen evolucionar la población en base a la presión que ejercen los operadores de selección, cruce y mutación.

A diferencia de los algoritmos evolutivos, en los cuales el concepto de memoria en la optimización no existe como tal, limitándose en el caso de aplicar elitismo, a seguir de forma muy pausada las tendencias del mejor individuo, existen otros métodos heurísticos que por sus principios han experimentado un auge considerable en los últimos años, aunando una mayor facilidad para ajustar el algoritmo con una interacción entre soluciones que, dependiendo del problema al que se apliquen, puede llegar a acelerar considerablemente la convergencia. Entre estos métodos destaca la optimización de enjambre de partículas, más conocido como *particle swarm optimization*. Introducido como método de optimización por Kennedy y Eberhart (1995), este método estocástico de optimización global se basa en imitar a nivel computacional el comportamiento de un colectivo a partir de la interacción entre sus miembros y con el entorno en el que éstos se desenvuelven.

El término enjambre o *swarm* hace referencia a una colección de agentes, individuos o partículas, a los que se les atribuye una memoria y una capacidad de organizarse y cooperar entre sí. Los ejemplos más claros lo constituyen las abejas en su búsqueda de alimentos alrededor de la colmena, las bandadas de aves, el sistema inmune,

que es en realidad un conjunto de células y moléculas, e incluso una muchedumbre puede verse como un grupo de personas que comparten impresiones para tomar decisiones, aprovechándose de los logros de sus congéneres y de su propia experiencia.

A diferencia de los algoritmos evolutivos, en la optimización de enjambre de partículas la población tiene memoria, es decir, la optimización se dirige y encauza influida por la historia pasada, por la memoria de cada individuo y por el estado presente en el que cada uno se encuentra. Si a esto se le une el hecho de utilizar un único operador con un número de parámetros a sintonizar muy reducido, queda justificado el reciente éxito que esta técnica de optimización está adquiriendo diferentes aplicaciones.

Con matices, pero bajo unos principios similares, Dorigo y otros (1996) proponen la así denominada optimización de colonias de hormigas o *ant colony optimization*. Básicamente, los principios del método se limitan a imitar el desplazamiento de las hormigas sobre lo que ahora es un espacio de soluciones, teniendo en cuenta que en su desplazamiento las hormigas trazan unos caminos de feromona que se disipa con el tiempo y la distancia. Evidentemente, en un cierto punto la intensidad de feromona es mayor cuanto mayor número de hormigas pasan por dicho punto o si éste ha sido visitado recientemente. Como resultado y siguiendo estas trayectorias, las hormigas se congregarán entorno a una cierta región del espacio en la que se encuentra la solución del problema.

Otros algoritmos como los algoritmos meméticos (*Memetic Algorithms*) y los algoritmos culturales (*Cultural Algorithms*) se distancian ligeramente de los principios asociados con la llamada inteligencia del enjambre (*swarm intelligence*), pero, en realidad, mantienen la idea de imitar y aplicar procesos naturales a la optimización. Los algoritmos meméticos, introducidos por Moscazo (1989), combinan una estrategia basada en población con una búsqueda local. A grandes rasgos y a diferencia de los algoritmos evolutivos, los algoritmos meméticos intentan imitar la evolución cultural de un colectivo en lugar de su evolución biológica.

Por otra parte y aprovechando las afirmaciones vertidas por diversos sociólogos que sugieren que la cultura puede ser simbólicamente codificada y transmitida entre generaciones como un mecanismo más de herencia, Reynolds (1994) propone un modelo computacional que da lugar a los así denominados algoritmos culturales. Los algoritmos

culturales se diferencian de los evolutivos por el hecho de poseer memoria, de tal forma que la población mantiene una memoria de grupo o espacio de opinión con información de las soluciones potencialmente mejores y también de aquellas peores, con el objeto de dirigir la búsqueda. Básicamente, en los algoritmos culturales hay dos clases de información hereditaria entre generaciones, una basada en la transmisión de los comportamientos entre individuos y otra que contempla la formación de opiniones en función de las experiencias individuales.

A continuación se explican más detalladamente los métodos de optimización que han sido utilizados para desambiguación del sentido de las palabras: Templado Simulado y Algoritmos Genéticos. También explicamos los Algoritmos con Estimación de Distribuciones (*Estimation of Distribution Algorithms* o EDA por sus siglas en inglés), ya que en esta tesis se aplican estos algoritmos para la desambiguación del sentido de las palabras.

4.3 Templado simulado (Simulated Annealing)

El método de templado simulado es una técnica para la resolución de problemas de optimización combinatoria a gran escala. El nombre de este algoritmo es una analogía del proceso metalúrgico en el cuál, el metal se enfría y se templa. La característica de este fenómeno es que en el enfriamiento lento alcanza una composición uniforme y un estado de energía mínimo, sin embargo, cuando el proceso de enfriamiento es rápido, el metal alcanza un estado amorfo y con un estado alto de energía. En templado simulado la variable **T** corresponde a la temperatura que decrece lentamente hasta encontrar el estado mínimo.

El proceso requiere una función ***E***, la cual representa el estado de energía de cada configuración del sistema. Es esta función la que se intenta minimizar. A grandes rasgos el algoritmo funciona de la siguiente manera: se selecciona un punto inicial y además se escoge otra configuración de manera aleatoria, se calcula para ambas configuraciones su valor ***E***, si el nuevo valor es menor que el seleccionado como punto inicial, entonces el inicial es remplazado por la nueva configuración. Una característica esencial del

templado simulado es que, existe el caso en el que la nueva configuración es mayor a la configuración obtenida anteriormente, y la nueva es seleccionada. Esta decisión es tomada de manera probabilística y permite salir de algún mínimo local. Una vez que el método mantenga la misma configuración por un determinado tiempo, dicha configuración es escogida como la solución.

Cowie *et al.* (1992) utilizó este método para desambiguación de sentidos de palabras de la siguiente forma:

1. El algoritmo define una función **E** para la combinación de sentidos de palabras en un texto dado.
    1. Se calcula ***E*** para la configuración inicial **C**, donde **C** es el sentido más frecuente para cada palabra.
    2. Para cada iteración, se escoge aleatoriamente otra configuración conocida como **C'**, y se calcula su valor de ***E***. Si el valor de ***E*** para ***C'*** es menor que el de ***C*** entonces se elige ***C'*** como configuración inicial.
    3. La rutina termina cuando la configuración de sentidos no ha cambiado en un tiempo determinado.

4.4 Algoritmos genéticos (Genetic Algorithms)

Introducidos por Holland (1975) e impulsados en años sucesivos por Goldberg (1989), uno de sus estudiantes, los algoritmos genéticos han sido utilizados con éxito en múltiples campos de la ciencia. Los algoritmos genéticos son métodos sistemáticos para la resolución de problemas de *búsqueda y optimización*, que aplican a éstos los mismos métodos de la evolución biológica: selección basada en la población, reproducción sexual y mutación.

En un algoritmo genético, tras parametrizar el problema en una serie de variables, $(x_i,...,x_n)$ se codifican en un cromosoma. Todos los operadores utilizados por un algoritmo genético se aplicarán sobre estos cromosomas, o sobre poblaciones de ellos. En el algoritmo genético va implícito el método para resolver el problema; son solo parámetros de tal método los que están codificados, a diferencia de otros algoritmos evolutivos como

la programación genética. Hay que tener en cuenta que un algoritmo genético es independiente del problema, lo cual lo hace un algoritmo *robusto*, por ser útil para cualquier problema, pero a la vez *débil*, pues no está especializado en ninguno.

Las soluciones codificadas en un cromosoma *compiten* para ver cuál constituye la mejor solución (aunque no necesariamente la mejor de todas las soluciones posibles). El *ambiente,* constituido por otras soluciones, ejercerá una presión selectiva sobre la población, de forma que sólo los mejor adaptados (aquellos que resuelvan mejor el problema) sobrevivan o leguen su material genético a las siguientes generaciones, igual que en la evolución de las especies. La diversidad genética se introduce mediante mutaciones y reproducción sexual.

En la naturaleza lo único que hay que optimizar es la supervivencia, y eso significa a su vez maximizar diversos factores y minimizar otros. Un algoritmo genético, sin embargo, se usará habitualmente para optimizar sólo una función, no diversas funciones relacionadas entre sí simultáneamente. La optimización que busca diferentes objetivos simultáneamente, denominada multimodal o multiobjetivo, también se suele abordar con un algoritmo genético especializado.

Por lo tanto, un algoritmo genético consiste en lo siguiente: hallar de qué parámetros depende el problema, codificarlos en un cromosoma, y se aplican los métodos de la evolución: selección y reproducción sexual con intercambio de información y alteraciones que generan diversidad.

Gelbukh *et al.* (2003b) utilizaron un algoritmo genético que elige los sentidos que dan más coherencia al texto en términos de medidas de relación de palabras. El método optimiza globalmente el total de relaciones de palabras y no cada palabra de manera independiente.

4.5 Algoritmos con estimación de distribuciones (EDA)

Los EDA son algoritmos heurísticos de optimización que basan su búsqueda, al igual que los algoritmos genéticos, en el carácter estocástico de la misma y también se basan en

poblaciones que evolucionan. Sin embargo, a diferencia de los algoritmos genéticos, la evolución de las poblaciones no se lleva a cabo por medio de los operadores de cruce y mutación. En lugar de ello la nueva población de individuos se muestrea de una distribución de probabilidad, la cual es estimada de la base de datos conteniendo al conjunto de individuos seleccionados de entre los que constituyen la generación anterior.

Las características principales de los algoritmos con estimación de distribuciones son:

2. Basada en poblaciones
3. Sin operadores de cruce ni mutación
4. En cada generación se estima de los individuos seleccionados, la distribución de probabilidad subyacente a los mismos
5. Muestreando esta distribución se obtiene la siguiente población
6. Se repiten los dos pasos anteriores hasta el criterio de terminación

En la siguiente figura podemos observar la estructura general de un algoritmo con estimación de distribuciones.

```
1.- D_0 <- Generar la población inicial (m individuos)
2.- Evaluar la población D_0; k=1
3.- Repetir hasta condición de parada
3.1.- D_{k-1} <- Seleccionar n < m individuos de D_{k-1}
3.2.- Estimar un nuevo modelo M a partir de D_{k-1}
3.3.- D_{k-1}_m <- Muestrear m individuos a partir de M
3.4.- Evaluar D_{k-1}
3.5.- D_k <- Seleccionar m individuos de D_{k-1} unión D_{k-1}_m
```

Figura 8. Estructura General de un EDA

Como se puede ver, al igual que en la mayoría de los algoritmos generacionales, se parte de una población inicial con m individuos, generada (en la mayoría de los casos) aleatoriamente. En el segundo paso, un número n menor que m de individuos se seleccionan (normalmente aquellos con los mejores valores en cuanto a la función de evaluación) como base de datos para la estimación del modelo. A continuación se induce el modelo probabilística n-dimensional que mejor refleja las interdependencias entre las n variables. A partir del modelo inducido se genera una población auxiliar mediante muestreo. Por último, la nueva población D_i se obtiene a partir de la población anterior D_{i-1} y de la población auxiliar. Normalmente, esta selección se realiza de forma elitista.

El principal problema que se presenta es la estimación del modelo M, ya que cuanto más complejo sea el modelo, mejor recogerá las dependencias entre variables, pero más compleja será su estimación.

4.6 Conclusiones

En este capítulo vimos que la optimización puede definirse como aquella ciencia encargada de determinar las mejores soluciones a problemas matemáticos que a menudo modelan una realidad. La idea principal es maximizar o minimizar un criterio determinado.

Hay dos tipos principales de optimización: local y global. Los métodos locales obtienen la mejor solución posible en las inmediaciones del punto inicial, atribuyéndoseles una fuerte dependencia del punto de arranque del algoritmo. Por otra parte, las técnicas de optimización global exhiben una gran independencia de la naturaleza del espacio de soluciones y, a diferencia de las técnicas de búsqueda local, son capaces de atravesar un espacio de búsqueda con múltiples mínimos o máximos locales y alcanzar una solución global al problema, entendiendo como tal la mejor solución posible o una solución en las inmediaciones de la región que contiene a la solución óptima.

Vimos también una clasificación de los métodos de optimización basados en las heurísticas tradicionales y heurísticas modernas. Estas últimas resultan ser más eficientes que las primeras.

Estudiamos a detalle los métodos de optimización global que han sido utilizados en desambiguación del sentido de las palabras como: genéticos y templado simulado. También estudiamos los algoritmos con estimación de distribuciones que se utilizan en esta tesis.

CAPÍTULO

5

Propuesta

Capítulo 5. Propuesta

5.1 Introducción

Con base en los resultados reportados en el estado del arte para Lesk completo y Lesk simple, los autores han mostrado que el método simplificado de Lesk tiene mejores resultados en comparación con el método de Lesk completo. A diferencia de ellos, nosotros planteamos la hipótesis de que el algoritmo de Lesk completo es mejor que el algoritmo de Lesk simple, solo que no se han utilizado los métodos de optimización adecuados.

En este capítulo se propone el uso de un método de optimización, que no ha sido utilizado en la tarea de WSD, para mejorar los resultados de Lesk completo en comparación con los que se han obtenido previamente con otros métodos de optimización. Por otro lado, se propone utilizar este mismo método de optimización para demostrar que el algoritmo de Lesk completo (optimizado) es mejor que el algoritmo simplificado de Lesk, a diferencia de lo que se ha creído antes.

Se proponen dos nuevos algoritmos para la desambiguación del sentido de las palabras basados en los métodos tipo Lesk. El primer método propuesto es la combinación de dos algoritmos: Lesk simple y Lesk optimizado. El segundo es una modificación del algoritmo simplificado de Lesk.

5.2 Lesk completo usando algoritmos con estimación de distribuciones

La principal desventaja del algoritmo original de Lesk (1986) es su complejidad. Para aliviar este problema existen dos alternativas: a) El uso de métodos de optimización para encontrar la combinación de sentidos cercana al óptimo real, y b) Una variación del algoritmo de Lesk conocida como Lesk simple.

Entre los algoritmos de optimización usados para la desambiguación del sentido de las palabras se encuentran templado simulado (Cowie *et al.* 1992) y algoritmos genéticos (Gelbukh *et al.* 2003b). Basados en los resultados reportados en el estado del arte sobre el uso de estos métodos para WSD, proponemos el uso de un algoritmo de optimización distinto conocido como algoritmo con estimación de distribuciones (EDA), para mejorar los resultados del algoritmo de Lesk, en comparación con los resultados que se han obtenido para templado simulado o algoritmos genéticos en la tarea de WSD.

Seleccionamos los algoritmos con estimación de distribuciones por dos razones principales: 1) Son algoritmos rápidos, y, 2) Su efectividad no está fuertemente correlacionada con los parámetros de inicialización.

5.3 Lesk completo usando EDA vs. Lesk simple

La otra alternativa mencionada arriba, para aliviar el problema de la complejidad del algoritmo de Lesk original es el algoritmo simplificado de Lesk. El estado del arte muestra que Lesk simple tiene mejores resultados en comparación con Lesk completo (optimizado con los métodos mencionados en la sección anterior). Por lo tanto, para demostrar nuestra hipótesis de que, el algoritmo de Lesk completo con un método de optimización más adecuado tiene mejores resultados en comparación con el algoritmo simplificado de Lesk, proponemos el uso de los algoritmos con estimación de distribuciones para el método de Lesk completo en la desambiguación del sentido de las palabras.

5.4 Lesk simple con *back-off* a Lesk optimizado

Haciendo un análisis de los resultados obtenidos para Lesk optimizado y Lesk simple sin estrategia de *back-off* para la variante de glosa y ejemplos (Sección 6.4), podemos observar que la cobertura (porcentaje de casos en los que el método toma decisión) del algoritmo de Lesk simple es mucho menor (18.6%) en comparación con la cobertura del algoritmo de Lesk optimizado (91.2%). Sin embargo, la precisión (porcentaje de casos en los que el método elige correctamente) de Lesk simple es mayor (52%) en comparación a la precisión que tiene Lesk optimizado (44.2%).

Por lo tanto se hizo, como propuesta de algoritmo para desambiguación del sentido de las palabras, una combinación de los algoritmos Lesk simple y Lesk optimizado con EDA, dicha combinación consiste en dejar que un método elija primero, y sobre los casos que no eligió, que decida el otro método. Se decidió que el método que elija primero es el de Lesk simple, debido a dos situaciones: a) la cobertura de Lesk simple es muy baja, y b) la precisión de Lesk simple es mejor que la de Lesk completo optimizado. De tal forma, dejamos que Lesk simple elija (con más precisión) en los pocos casos que cubre y, en el resto de los casos, elige Lesk optimizado con EDA.

5.5 Modificación al algoritmo simplificado de Lesk

Haciendo un análisis de los resultados del algoritmo de Lesk simple, la cobertura y precisión del método, cuando se utiliza la glosa y ejemplos (juntos) como definición del sentido de una palabra, es mayor con respecto a cuando se utilizan por separado (glosa sola o ejemplos solos). Esto parece indicar que, mientras más información tiene Lesk simple para elegir un sentido, mayor cobertura y precisión tiene, por lo tanto se decidió implementar una modificación para mejorar el desempeño de este método.

El método de Lesk simple consiste en elegir el sentido de una palabra a la vez, contando el número de palabras en común (traslapes) que contiene cada sentido

(definición) de la palabra ambigua y sus palabras vecinas (contexto); el sentido elegido es aquel que contiene más traslapes.

La modificación propuesta es, crear una bolsa de palabras que contiene las definiciones de todos los sentidos de las palabras del contexto de la palabra a desambiguar, y al igual que Lesk simple, elegir el sentido de una palabra contando el número de traslapes que contiene cada sentido (definición) de la palabra ambigua y la bolsa de palabras, dándole así más información al método para tomar una decisión.

5.6 Conclusiones

En este capítulo se describieron las propuestas para mejorar el desempeño de los métodos tipo Lesk para desambiguación del sentido de las palabras, que son:

1) Uso de algoritmos con estimación de distribuciones, como método de optimización, aplicado al algoritmo de Lesk completo para desambiguación del sentido de las palabras, con el fin de mejorar los resultados obtenidos con otros métodos de optimización.
2) Uso de algoritmos con estimación de distribuciones, como método de optimización, aplicado al algoritmo de Lesk completo para desambiguación del sentido de las palabras, con el fin de demostrar que Lesk completo, con un método más adecuado de optimización tiene mejores resultados en comparación con los de Lesk simple.
3) Nuevo algoritmo para WSD basado en la unión del método de Lesk simple con el método de Lesk completo optimizado con EDA.
4) Nuevo algoritmo para WSD basado en una modificación al algoritmo simplificado de Lesk.

CAPÍTULO

6

Resultados Experimentales

Capítulo 6. Resultados experimentales

6.1 Introducción

En este capítulo se describe la metodología experimental de este trabajo para llevar a cabo la evaluación de los métodos de desambiguación del sentido de las palabras basados en la aplicación directa del diccionario: Lesk completo optimizado y Lesk simple; dentro de esta metodología, se describen los parámetros del método de optimización utilizado en este trabajo: Algoritmos con Estimación de Distribuciones.

Se describen los resultados para el algoritmo de Lesk optimizado usando algoritmos con estimación de distribuciones; así como los resultados para el algoritmo de Lesk simple. Los resultados se muestran sobre los datos de Senseval-2 y están dados para diferentes estrategias de *back-off*.

Además se presentan los resultados obtenidos para los dos nuevos algoritmos para WSD propuestos y descritos en el capítulo anterior.

Se presenta un análisis detallado de los resultados para métodos tipo Lesk, y la interacción de éstos con diferentes estrategias de *back-off*. Dentro de este análisis se presenta la comparación de los resultados obtenidos para cada una de las propuestas de este trabajo con los resultados del estado del arte.

6.2 Metodología de experimentación

En este trabajo se utilizaron como algoritmos para desambiguación de sentidos de palabras Lesk completo y Lesk simple, además del uso de dos estrategias de *back-off*; sentido aleatorio y sentido más frecuente.

Para llevar a cabo la desambiguación, los métodos tipo Lesk requieren el uso de un diccionario de sentidos, en nuestro caso, el diccionario utilizado fue WordNet.

La evaluación se hizo sobre el corpus etiquetado sintáctica y semánticamente Senseval-2 English all-words.

La medida de similitud utilizada por ambos algoritmos es la medida original de Lesk, traslape.

La ventana de contexto para ambos algoritmos es la oración.

Para la optimización del método de Lesk completo se utilizó un método de optimización conocido como Algoritmos con Estimación de Distribuciones, los parámetros de dicho algoritmo se presentan a continuación.

6.2.1 Parámetros del algoritmo con estimación de distribuciones

El algoritmo con estimación de distribuciones está implementado en MatLab y los parámetros del algoritmo son:

1.- Sea S un conjunto de enteros, en nuestro caso las palabras; con A=|S|, es decir A=número de palabras en la oración; y S[i] un elemento del conjunto S, i={1,2,...,A}.

Ejemplo: S = {149,340,101,289,160,33,320}; A = 7, S[1] = 149, S[2] = 340 ...

2.- El espacio de búsqueda: Sea C un cubo (discreto) de dimensión N. La dimensión del espacio de búsqueda de C (al que llamaremos W) está dada por la multiplicación de n(S[i]), donde n es una función que determina el número de sentidos de una palabra.

Ejemplo: Si n(149) = 3, n(340) = 1, n(101) = 6, ..., n(320) = 5, entonces el espacio de búsqueda en C es un cubo discreto de 3 x 1 x 6 x ... x

3.- La función de costo F(p) de un punto específico "p"(combinación de sentidos) en "C" es calculado como F(p) = Σ R(S[i],p[i],S[j],p[j]) con i,j= {1,2...,N} i>j, donde R es una función dada (relación de similitud entre 2 pares de sentidos) y nuestros datos 0<=R(...)<1. En la práctica, esta función es "escasa", es decir, en muchos casos R = 0;

Ejemplo: Dado p = (2,1,5,...,3) para el cubo C, suponga que
R(149,2,340,1) = 0.123, R(149,2,101,5) = 0.456,R (340, 1, 101, 5) =
0.789, Tenemos entonces F(p) = 0.123 + 0.456 + 0.789 + ...

4.- Se requiere encontrar "p"(combinación de sentidos) que MAXIMIZA la función de costo F.

El algoritmo entonces:

1. Genera la población inicial (D_0) de m individuos (combinación de sentidos) aleatoriamente. m<W.

2. Calcula la función de costo F(p) para la población inicial D_0 y k=1

3. Repite, hasta que, después de varias generaciones los nuevos individuos no mejoran con respecto al mejor de los individuos obtenidos en las generaciones previas
   a) Selecciona t < m individuos de la población anterior (`D_{k-1})(los que tienen` mejores valores en cuanto a la función de evaluación F(p)
   b) Para la nueva población se lleva a cabo la inducción del modelo probabilístico n-dimensional que mejor refleja las interdependencias entre las n variables
   c) Se obtiene la nueva población seleccionando m nuevos individuos por medio de la simulación de la distribución de probabilidad aprendida en el paso anterior
   d) Se evalúa la función de costo para la nueva población

6.3 Resultados obtenidos

A continuación se presentan los resultados para Lesk optimizado (usando EDAs), Lesk simple y los nuevos algoritmos propuestos: Lesk simple con back-off a lesk optimizado y Lesk simple modificado.

La medida de similitud usada fue la de Lesk original, es decir, traslape de palabras entre los sentidos.

La descripción del sentido de una palabra consiste de glosa (descripción misma) y ejemplos. Generalmente, cuando se evalúan los métodos tipo Lesk para la desambiguación de sentidos se utiliza la glosa y ejemplos juntos como la descripción de sentido de una palabra. En nuestro caso, para la evaluación de Lesk optimizado y Lesk simple consideramos 3 variantes: glosa sola, ejemplos solos, glosa y ejemplos.

Para cada variante, además de evaluar el método por sí solo, se consideró evaluarlo con dos diferentes estrategias de *back-off*: sentido aleatorio y sentido más frecuente. A continuación se muestran los resultados obtenidos.

6.3.1 Preprocesamiento de datos

Senseval-2 contiene tres archivos con 2436 palabras ambiguas en 238 oraciones, de las cuales 1136 son sustantivos, 544 verbos, 457 adjetivos y 299 adverbios. La tabla 9 muestra las categorías gramaticales de las palabras para cada uno de los archivos.

Tabla 9. Categoría gramatical para las palabras (datos depurados) de Senseval-2

ARCHIVO	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
Archivo 1 (D00)	99	86	331	162
Archivo 2 (D01)	170	107	495	242
Archivo 3 (D02)	188	106	310	140

Para llevar a cabo la desambiguación fue necesario eliminar del corpus:

1) Oraciones con una sola palabra
2) Oraciones que contenían 2 palabras y una de ellas sin sentidos según WordNet (ejemplo: n't)
3) Palabras que no tienen sentido según WordNet
4) Palabras etiquetadas en el corpus como "desconocidas"

Quedando los datos de la siguiente manera:

Tres archivos con 2377 palabras en 236 oraciones, de las cuales 1112 son sustantivos, 518 verbos, 448 adjetivos y 299 adverbios. La tabla 10 muestra las categorías gramaticales para las palabras de estos datos:

Tabla 10. Categoría gramatical para las palabras de los datos depurados de Senseval-2.

ARCHIVO	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
Archivo 1 (D00)	96	86	329	160
Archivo 2 (D01)	167	107	484	229
Archivo 3 (D02)	185	106	299	129

6.3.2 Resultados para el método de Lesk optimizado usando algoritmos con estimación de distribuciones

6.3.2.1 Sin ninguna estrategia de *back-off*

La tabla siguiente muestra los resultados de *coverage* (C) , *precision* (P) y *recall* (R) para tres variantes del método de desambiguación Lesk optimizado sin ninguna estrategia de *back-off*, es decir, el método por sí solo; evaluado sobre los datos de Senseval-2: a) Glosa, b) Ejemplos y c) Glosa y Ejemplos. Estas variantes se refieren a qué se considera como definición de sentido de una palabra al momento de comparar.

Tabla 11. Resultados obtenidos para Lesk optimizado sin ninguna estrategia de back-off

	Glosa			*Ejemplos*			*Glosa y Ejemplos*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	83.8	37.3	31.3	63.3	29.1	18.4	92.1	39.8	36.7
Archivo 2 (D01)	81.2	45.1	36.6	60.5	35.3	21.3	92.0	48.4	44.5
Archivo 3 (D02)	77.1	40.2	31.0	64.5	40.4	26.0	89.6	44.5	39.9
Promedio	*80.7*	*40.9*	*33.0*	*62.8*	*34.9*	*21.9*	***91.2***	***44.2***	***40.4***

6.3.2.2 Con *back-off* a sentido aleatorio

La tabla siguiente muestra los resultados de *coverage* (C) , *precision* (P) y *recall* (R) para tres variantes del método de desambiguación Lesk optimizado con *back-off* a sentido aleatorio, evaluado sobre los datos de Senseval-2: a) Glosa, b) Ejemplos y c) Glosa y

Ejemplos. Estas variantes se refieren a qué se considera como definición del sentido de una palabra al momento de comparar.

Tabla 12. Resultados obtenidos para Lesk optimizado con *back-off* a sentido aleatorio

	Glosa			*Ejemplos*			*Glosa y Ejemplos*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	99.0	42.1	41.7	99.0	39.9	39.5	99.0	42.8	42.4
Archivo 2 (D01)	97.6	48.5	47.3	97.6	45.7	44.6	97.6	49.9	48.7
Archivo 3 (D02)	96.7	44.9	43.5	96.7	47.7	46.1	96.7	46.7	45.1
Promedio	*97.8*	*45.2*	*44.2*	*97.8*	*44.4*	*43.4*	***97.8***	***46.5***	***45.4***

6.3.2.3 Con *back-off* a sentido más frecuente

En la tabla 13 se muestran los resultados de *coverage* (C) , *precision* (P) y *recall* (R) para tres variantes del método de desambiguación Lesk optimizado con *back-off* a sentido más frecuente, evaluado sobre los datos de Senseval-2: a) Glosa, b) Ejemplos y c) Glosa y Ejemplos. Estas variantes se refieren a qué se considera como definición de sentido de una palabra al momento de comparar.

Tabla 13. Resultados obtenidos para Lesk optimizado con back-off a sentido más frecuente

	Glosa			*Ejemplos*			*Glosa y Ejemplos*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	99.0	43.6	43.1	99.0	45.6	45.2	99.0	43.1	42.7
Archivo 2 (D01)	97.6	49.6	48.4	97.6	51.9	50.7	97.6	50.2	49.0
Archivo 3 (D02)	96.7	47.1	45.6	96.7	51.6	49.9	96.7	47.0	45.4
Promedio	*97.8*	*46.8*	*45.7*	***97.8***	***49.7***	***48.6***	*97.8*	*46.8*	*45.7*

6.3.3 Resultados para el método de Lesk simple

6.3.3.3 Sin ninguna estrategia de *back-off*

La tabla siguiente muestra los resultados de *coverage* (C) , *precision* (P) y *recall* (R) para tres variantes del método de desambiguación Lesk simple sin ninguna estrategia de *back-off*, es decir, el método por sí solo; evaluado sobre los datos de Senseval-2: a) Glosa, b)

Ejemplos y c) Glosa y Ejemplos. Estas variantes se refieren a qué se considera como definición de sentido de una palabra al momento de comparar.

Tabla 14. Resultados obtenidos para Lesk simple sin ninguna estrategia de back-off

	Glosa			*Ejemplos*			*Glosa y Ejemplos*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	11.8	55.6	6.6	13.2	36.7	4.8	19.3	47.7	9.2
Archivo 2 (D01)	10.9	60.2	6.6	11.3	53.8	6.1	18.9	59.5	11.2
Archivo 3 (D02)	10.7	34.6	3.7	11.5	55.2	6.3	17.8	48.9	8.7
Promedio	*11.1*	*50.1*	*5.6*	*12.0*	*48.6*	*5.7*	***18.6***	***52.0***	***9.7***

6.3.3.2 Con *back-off* a sentido aleatorio

La tabla 15 muestra los resultados de *coverage* (C) , *precision* (P) y *recall* (R) para tres variantes del método de desambiguación Lesk simple con *back-off* a sentido aleatorio, evaluado sobre los datos de Senseval-2: a) Glosa, b) Ejemplos y c) Glosa y Ejemplos. Estas variantes se refieren a qué se considera como definición de sentido de una palabra al momento de comparar.

Tabla 15. Resultados obtenidos para Lesk simple con back-off a sentido aleatorio

	Glosa			*Ejemplos*			*Glosa y Ejemplos*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	99.0	38.4	38.0	99.0	36.5	36.1	99.0	40.8	40.4
Archivo 2 (D01)	97.6	42.2	41.2	97.6	43.3	42.2	97.6	44.3	43.2
Archivo 3 (D02)	96.7	42.5	41.1	96.7	40.4	39.1	96.7	45.5	44.0
Promedio	*97.8*	*41.0*	*40.1*	*97.8*	*40.1*	*39.1*	***97.8***	***43.5***	***42.5***

6.3.3.3 Con *back-off* a sentido más frecuente

En la tabla 16 se muestran los resultados de *coverage* (C) , *precision* (P) y *recall* (R) para tres variantes del método de desambiguación Lesk simple con *back-off* a sentido más frecuente, evaluado sobre los datos de Senseval-2: a) Glosa, b) Ejemplos y c) Glosa y Ejemplos. Estas variantes se refieren a qué se considera como definición de sentido de una palabra al momento de comparar.

Tabla 16. Resultados obtenidos para Lesk simple con back-off a sentido más frecuente

	Glosa			*Ejemplos*			*Glosa y Ejemplos*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	99.0	59.4	58.8	99.0	58.6	58.0	99.0	59.7	59.1
Archivo 2 (D01)	97.6	61.4	59.9	97.6	63.0	61.4	97.6	62.0	63.5
Archivo 3 (D02)	96.7	58.2	56.3	96.7	61.9	59.8	96.7	60.1	58.1
Promedio	*97.8*	*59.7*	*58.3*	***97.8***	***61.2***	***59.7***	*97.8*	*60.6*	*60.2*

6.3.4 Resultados para el método propuesto de Lesk simple con *back-off* a Lesk optimizado

A continuación se presentan los resultados obtenidos para este método sobre los datos de Senseval-2. En la tabla siguiente, además de los resultados de la combinación de los métodos (*LS+LC*), se muestran los resultados obtenidos para este método sumando una tercer estrategia de *back-off*, sentido aleatorio (*LS+LC+SA*) y sentido más frecuente (*LS+LC+SMF*).

Tabla 17. Resultados obtenidos para el método propuesto
"Lesk simple con *back-off* a Lesk optimizado"

	LS+LC			*LS+LC+SA*			*LS+LC+SMF*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	92.1	40.0	36.8	99.0	42.8	42.4	99.0	43.3	42.8
Archivo 2 (D01)	92.2	49.1	42.3	97.6	50.1	48.9	97.6	50.8	49.6
Archivo 3 (D02)	89.7	43.7	39.2	96.7	45.4	43.9	96.7	46.2	44.6
Promedio	*91.3*	*44.3*	*39.4*	*97.8*	*46.1*	*45.1*	***97.8***	***46.8***	***45.7***

6.3.5 Resultados para el método propuesto de Lesk simple modificado

La evaluación del método se realizó sobre los datos de Senseval-2. La tabla 18 muestra los resultados obtenidos para Lesk simple modificado usando dos diferentes estrategias de *back-off*: sentido aleatorio (*back-off* a SA) y sentido más frecuente (*back-off* a SMF); y también sin ningún *back-off* (LS modificado).

Tabla 18. Resultados obtenidos para el método propuesto "Lesk simple modificado"

	LS modificado			*back-off a SA*			*back-off a SMF*		
	C	P	R	C	P	R	C	P	R
Archivo 1 (D00)	66.4	44.9	29.8	99.0	45.5	45.0	99.0	50.2	49.7
Archivo 2 (D01)	72.6	55.6	40.1	97.6	49.8	48.5	97.6	55.5	54.2
Archivo 3 (D02)	67.9	50.8	34.5	96.7	49.3	47.7	96.7	53.8	52.0
Promedio	*69.0*	*50.4*	*34.8*	*97.8*	*48.2*	*47.1*	***97.8***	***53.2***	***52.0***

6.4 Análisis de los resultados

Para llevar a cabo un análisis detallado de los resultados obtenidos en este trabajo para Lesk optimizado con EDA, Lesk simple, Lesk simple con back-off a Lesk optimizado y Lesk simple modificado; así como una comparación de estos métodos y su interacción con estrategias de *back-off*, la tabla siguiente muestra un condensado de los resultados obtenidos para cada método.

Tabla 19. Condensado de resultados

	ESTRATEGIA DE *BACK-OFF*	C	P	R
BASELINE: ALEATORIO	-	97.8	37.5	36.6
BASELINE: SENTIDO MÁS FRECUENTE	-	97.8	61.6	60.2
LESK OPTIMIZADO USANDO EJEMPLOS	*Aleatorio*	97.8	44.4	43.4
	Sentido más frecuente	97.8	49.7	48.6
	Sin back-off	62.8	34.9	21.9
LESK OPTIMIZADO USANDO GLOSA	*Aleatorio*	97.8	45.2	44.2
	Sentido más frecuente	97.8	46.8	45.7
	Sin back-off	80.7	40.9	33.0
LESK OPTIMIZADO USANDO GLOSA Y EJEMPLOS	*Aleatorio*	97.8	46.5	45.4
	Sentido más frecuente	97.8	46.8	45.7
	Sin back-off	91.2	44.2	40.4
LESK SIMPLE USANDO EJEMPLOS	*Aleatorio*	97.8	40.1	39.1
	Sentido más frecuente	97.8	61.2	59.7
	Sin back-off	12.0	48.6	5.7
LESK SIMPLE USANDO GLOSA	*Aleatorio*	97.8	41.0	40.1
	Sentido más frecuente	97.8	59.7	58.3
	Sin back-off	11.1	50.1	5.6

LESK SIMPLE USANDO GLOSA Y EJEMPLOS	*Aleatorio*	97.8	43.5	42.5
	Sentido más frecuente	97.8	60.6	60.2
	Sin back-off	18.6	52.0	9.7
LESK SIMPLE CON *BACK-OFF* A LESK OPTIMIZADO	*Aleatorio*	97.8	46.1	45.1
	Sentido más frecuente	97.8	46.8	45.7
	Sin back-off	91.3	44.3	39.4
LESK SIMPLE MODIFICADO	*Aleatorio*	97.8	48.2	47.1
	Sentido más frecuente	97.8	53.2	52.0
	Sin back-off	69.0	50.4	34.8

Como podemos ver en la tabla 19, los métodos sin *back-off* o con *back-off* a sentido aleatorio muestran mejores resultados cuando se utiliza glosa y ejemplos como definición del sentido de una palabra, esto sucede debido a que al usar glosa y ejemplos juntos le damos más información al método para tomar una decisión, aumentando la cobertura e incluso la precisión con respecto al uso de glosa o ejemplos por separado. De forma contraria, los métodos que utilizan *back-off* a sentido más frecuente muestran mejores resultados cuando sólo se utilizan los ejemplos, esto se debe a que usando solamente ejemplos en las definiciones de los sentidos hay menos información para que el sistema tome una decisión, de forma que la cobertura del método disminuye y el método de *back-off* decide más veces; como en estos casos el *back-off* es sentido más frecuente, la precisión del método aumenta debido a que el *baseline* sentido más frecuente tiene una precisión muy alta.

Por otro lado, sobre la interacción que tienen los métodos tipo Lesk para desambiguación del sentido de las palabras con las estrategias de *back-off* podemos decir que, un método aunado a una estrategia de *back-off* es bueno cuando:

a) La cobertura y precisión del método principal son buenas, de tal forma que la estrategia de *back-off* ya no importa.

b) La cobertura del método principal es mala y la precisión del *back-off* es buena, entonces la precisión del método principal no importa.

Por lo tanto, el buen desempeño de un método aunado a una estrategia de *back-off* no indica que el método es bueno, podría indicar que el método tiene mala cobertura, es decir, mientras peor sea la cobertura del método, mejores resultados tendrá la unión de éste con su *back-off*. En este caso se encuentra el método de Lesk simple, el cual tiene

muy baja cobertura, es decir, elige muy pocas veces, dando oportunidad al *back-off* de elegir la mayoría de los casos; por lo tanto, si se usa sentido más frecuente como estrategia de *back-off* para Lesk simple, el método aumenta por mucho la precisión, debido a que ésta es muy buena para sentido más frecuente.

Por el contrario, el método de Lesk optimizado tiene buena cobertura, de forma que la estrategia de *back-off* que se use no afecta en mucho el desempeño del método. Sin embargo, cabe señalar que agregar una estrategia de *back-off* (incluso no muy buena, como es sentido aleatorio) a un método que tiene buen desempeño, algunas veces ayuda, es decir, si el método cubre los casos difíciles para el *back-off*, entonces la precisión del *back-off* es mala sobre todos los casos pero no sobre los residuos del método principal. En este caso específico, Lesk optimizado cubre palabras con muchos sentidos porque tiene mayor probabilidad de tomar una decisión, dejando a *back-off* sentido aleatorio los casos con pocos sentidos (uno o dos), haciendo que la decisión sea mucho más fácil.

Al comparar Lesk optimizado vs. Lesk simple, podemos observar en la tabla 19, que Lesk optimizado es mejor cuando la estrategia de *back-off* es sentido aleatorio, mientras que Lesk simple tiene mejor desempeño cuando se usa con *back-off* a sentido más frecuente. Sobre este punto podemos decir que, a nuestro parecer, utilizar sentido más frecuente como estrategia de *back-off* no es correcto cuando hablamos de métodos no supervisados, como es el caso de los métodos (tipo Lesk) que estamos evaluando; la estrategia de *back-off* a sentido más frecuente es supervisada (Mihalcea 2005), de forma que al utilizarla se considerarían métodos supervisados. Por lo tanto, creemos que lo correcto es utilizar una estrategia de *back-off* que sea no supervisada, en nuestro caso, sentido aleatorio y de esa forma, podemos decir que Lesk optimizado muestra mejor desempeño en comparación con Lesk simple.

Comparando los resultados del método propuesto, Lesk simple con *back-off* a Lesk optimizado, con los resultados dados para Lesk simple y Lesk optimizado por separado tenemos que, los resultados obtenidos para el método propuesto son muy similares a los que presenta Lesk optimizado, esto sucede debido a la poca cobertura que presenta Lesk simple, haciendo que prácticamente Lesk optimizado elija la mayoría de

los casos. Sin embargo, los resultados de este método propuesto son mejores en comparación con los que muestra Lesk simple por separado.

Comparando los resultados del método propuesto, Lesk simple modificado, con los resultados dados para Lesk simple tenemos que, cuando la estrategia de *back-off* utilizada es sentido más frecuente, Lesk simple supera los resultados obtenidos por nuestro método; como se discutió antes, Lesk simple tiene baja cobertura y el *back-off* a sentido más frecuente buena precisión, haciendo que el método mejore.

Por otro lado, usando *back-off* a sentido aleatorio, el método propuesto (Lesk simple modificado) muestra mejores resultados en comparación a Lesk simple, lo cual nos dice que el método propuesto es mejor, debido a que al usar esta estrategia de *back-off* no se beneficia a métodos con baja cobertura.

6.5 Comparación de los resultados con el estado del arte

Vasilescu *et al.* (2004) evaluaron el algoritmo de Lesk original (exhaustivo) y Lesk simple, ambos con *back-off* a sentido más frecuente. La evaluación la hicieron sobre los datos de Senseval-2 "English-all words" y Semcor. La medida de similitud usada fue traslape y evaluaron diferentes ventanas de contexto: ±2, ±3, ±8, ±10 y ±25. En la tabla 20 sólo se muestran los resultados para ventana de contexto ±2.

Mihalcea (2005) evaluó el algoritmo de Lesk completo (optimizado con templado simulado) y Lesk simple, ambos con *back-off* a sentido aleatorio sobre tres corpus diferentes: Senseval-2, Senseval-3 y una muestra de 10 oraciones seleccionadas aleatoriamente de Semcor. La ventana de contexto fue oración y la medida de similitud fue traslape modificado (palabras en común de dos definiciones entre la longitud de cada definición).

La tabla 20 muestra los resultados de los métodos mencionados arriba, así como los resultados obtenidos para los métodos propuestos en esta tesis.

Tabla 20. Resultados reportados en el estado del arte y resultados de los métodos propuestos

	ESTRATEGIA DE BACK-OFF	Cobertura	Precisión	*Recall*
VASILESCU, ET AL. (2004) LESK COMPLETO (EXHAUSTIVO)	*Aleatorio*	-	-	-
	Sentido más frecuente	99.1	42.64	42.26
	Sin back-off	-	-	-
VASILESCU, ET AL. (2004) LESK SIMPLE	*Aleatorio*	-	-	-
	Sentido más frecuente	99.1	58.18	57.66
	Sin back-off	~ 22*	~ 45*	~ 9.9*
MIHALCEA (2005) LESK COMPLETO (TEMPLADO SIMULADO)	*Aleatorio*	100	39.5	39.5
	Sentido más frecuente	-	-	-
	Sin back-off	-	-	-
MIHALCEA (2005) LESK SIMPLE	*Aleatorio*	100	48.7	48.7
	Sentido más frecuente	-	-	-
	Sin back-off	-	-	-
LESK COMPLETO (EDA)	*Aleatorio*	97.8	46.5	45.4
	Sentido más frecuente	97.8	46.8	45.7
	Sin back-off	91.2	44.2	40.4
LESK SIMPLE	*Aleatorio*	97.8	43.5	42.5
	Sentido más frecuente	97.8	60.6	60.2
	Sin back-off	18.6	52.0	9.7
LESK SIMPLE CON *BACK-OFF* A LESK OPTIMIZADO	*Aleatorio*	97.8	46.1	45.1
	Sentido más frecuente	97.8	46.8	45.7
	Sin back-off	91.3	44.3	39.4
LESK SIMPLE MODIFICADO	*Aleatorio*	97.8	48.2	47.1
	Sentido más frecuente	97.8	53.2	52.0
	Sin back-off	69.0	50.4	34.8

Aunque no nos podemos comparar directamente con los resultados obtenidos por Vasilescu *et al.* (2004) dado que ellos utilizan una ventana de contexto de tamaño diferente al que utilizamos nosotros, podemos ver que:

1) Los resultados obtenidos por nosotros para el método de Lesk completo optimizado con EDA son mejores en comparación con los que ellos reportan para el método de Lesk completo exhaustivo.

2) Los resultados que obtuvimos para el método de Lesk simple son mejores en comparación con los reportados por Vasilescu *et al.* (2004) para este mismo método.

3) Comparando los resultados de Lesk optimizado con EDA obtenidos en este trabajo, con los de Lesk simple que ellos reportan, ambos con *back-off* a sentido más

* Datos calculados a partir de las gráficas de análisis de resultados de Vasilescu *et. al.* (2004)

frecuente, podemos observar que los resultados de Lesk simple son mejores. Cabe mencionar que en los resultados reportados, en esta tesis o en el estado del arte, referentes a los métodos con *back-off* a sentido más frecuente, Lesk simple siempre es mejor que Lesk completo. Esto se debe a la baja cobertura que presenta Lesk simple y la alta precisión del método de *back-off* a sentido más frecuente.

Comparando nuestros resultados con los reportados por Mihalcea (2005) podemos ver que:

1) Para el método de Lesk simple, ella reporta mejor precisión en comparación con nuestros resultados. Cabe hacer notar que, la única diferencia de sus experimentos con los nuestros es el uso de una medida de similitud diferente al traslape. Considerando esto, podemos decir que, el uso de esa medida de similitud le da una ventaja de precisión (comparando los resultados de Lesk simple). Por lo tanto, al comparar los resultados de Lesk completo usando EDA con los resultados que ella reporta para Lesk completo usando templado simulado, aún con la ventaja de la medida de similitud que utiliza, nuestros resultados son mejores.

2) Al comparar nuestros resultados de Lesk completo usando EDA con los resultado que Mihalcea reporta para Lesk simple podemos ver que, Lesk simple es mejor que Lesk completo usando EDA, pero esto sucede con la ventaja de precisión que le da el uso de la medida de similitud de traslape modificado. Podemos inferir que, si usáramos la medida de similitud que ella utiliza, muy probablemente nuestros resultados aumentarían, siendo así mejores que los que ella reporta.

6.6 Ventajas de los métodos propuestos

A continuación se enlistan las principales ventajas de cada uno de los métodos propuestos:

Lesk optimizado usando EDA

1) Mejores resultados en comparación con otros métodos de optimización usados para la tarea de desambiguación del sentido de las palabras.

2) Los algoritmos con estimación de distribuciones son mucho más rápidos que otros métodos de optimización, lo que reduce el tiempo de procesamiento.

3) Con el uso de los algoritmos con estimación de distribuciones se mejoran los resultados para el método de Lesk completo en comparación con los de Lesk simple.

Lesk simple con back-off a Lesk optimizado

1) La cobertura del método es 391% mayor en comparación con la cobertura que tiene el algoritmo de Lesk simple.

2) Este método muestra mejores resultados en comparación con los que tiene Lesk simple por separado.

Lesk simple modificado

1) La cobertura del método es 277% mayor en comparación con la cobertura que tiene el algoritmo de Lesk simple.

2) Mejores resultados en comparación con los que tiene Lesk simple

3) Mejores resultados en comparación con los que tiene Lesk completo optimizado con EDA.

6.7 Conclusiones

En este capítulo se describió la metodología utilizada para llevar a cabo evaluación sobre los métodos tipo Lesk para desambiguación del sentido de las palabras. Se presentaron los resultados para los métodos propuestos y se hizo un análisis sobre ellos.

Se presentó la comparación de los resultados obtenidos para los métodos propuestos con los resultados reportados en el estado del arte así como las ventajas principales de los métodos propuestos.

CAPÍTULO

7

Conclusiones y trabajo futuro

Capítulo 7. Conclusiones y trabajo futuro

7.1 Introducción

En el capítulo anterior se mostraron los resultados obtenidos en los experimentos realizados para desambiguación del sentido de las palabras basados en métodos tipo Lesk; utilizando algoritmos con estimación de distribuciones para Lesk completo. Basados en estos resultados, a continuación se presenta una pequeña discusión y las conclusiones derivadas de este trabajo.

Se presentan las aportaciones y trabajos publicados, y, por último, el trabajo futuro.

7.2 Discusión

Los resultados reportados por Lesk (1986) para su algoritmo fueron de 50-70% en ejemplos cortos.

Mihalcea (2005) evaluó Senseval-2 para Lesk simple y Lesk completo usando templado simulado, ambos con *back-off* a sentido aleatorio. Sus resultados fueron:

- Lesk completo: 39.5%
- Lesk simple: 48.7%

Además, Vasilescu *et al.* (2004) evaluaron Lesk simple y Lesk completo exhaustivo sobre Senseval-2, ambos métodos con *back-off* a sentido más frecuente con los siguientes resultados:

- Lesk completo: 42.6%

– Lesk simple: 58.2%

De ahí surgió la pregunta ¿Por qué los resultados de Lesk simple son mejores que los de Lesk original?, planteándonos una hipótesis: *Con un mejor algoritmo matemático de optimización, se pueden obtener buenos resultados en el método de Lesk, sin alterar su naturaleza lingüística*.

Así pues, para demostrar nuestra hipótesis, utilizamos un algoritmo con estimación de distribuciones para desambiguación del sentido de las palabras basándonos en el algoritmo de Lesk original (1986); evaluando éste sobre los datos de Senseval-2 en la tarea de "*English all-words*" y con diferentes estrategias de *back-off*: sentido aleatorio y sentido más frecuente.

Para llevar a cabo las comparaciones entre ambos métodos (optimizado y simple), evaluamos Lesk simple sobre los mismos datos y con las mismas estrategias de *back-off*.

Con el fin de tener una mejor comprensión sobre el desempeño de los algoritmos tipo Lesk para desambiguación del sentido de las palabras, se presentó un análisis de los resultados obtenidos, haciendo comparación de los métodos y explicando a detalle la interacción que tienen con diferentes estrategias de *back-off*. Con base en dicho análisis, se propusieron dos nuevos métodos para llevar a cabo la desambiguación del sentido de las palabras, basada en la aplicación directa del diccionario de sentidos.

En el primer método proponemos usar primero Lesk simple y después, en los casos en lo que éste no decide, usar Lesk optimizado. El segundo método propuesto es una modificación del algoritmo de Lesk simplificado y consiste en usar como contexto actual una bolsa de palabras que contiene las definiciones de todas las palabras vecinas a la palabra a desambiguar.

7.3 Conclusiones

A continuación se describen las conclusiones a las que llegamos con este trabajo:

1. El uso de más información en las definiciones del sentido de una palabra, se ve reflejado en el desempeño de los métodos para desambiguación del sentido de las palabras basados en la aplicación directa del diccionario de sentidos.
2. La estrategia de *back-off* influye en el rendimiento de los sistemas para desambiguación del sentido de las palabras.
3. Dos métodos en cadena pueden dar mejor resultado que por separado.
4. Una estrategia de *back-off* con buena precisión mejora el funcionamiento de un método con baja cobertura.
5. En la evaluación de métodos para desambiguación del sentido de las palabras, no se trata solo de precisión, sino también de cobertura.
6. El método de Lesk completo (optimizado con EDA) en comparación con Lesk simple es mejor cuando se utiliza una estrategia de *back-off* a sentido aleatorio.
7. El método de Lesk simple en comparación con Lesk optimizado es mejor cuando se utiliza con estrategia de *back-off* a sentido más frecuente, debido a la baja cobertura de Lesk simple y la alta precisión del método de *back-off* a sentido más frecuente.
8. Al evaluar métodos no supervisados para la desambiguación del sentido de las palabras no se debe utilizar sentido más frecuente como estrategia de *back-off*, ya que ésta es una técnica supervisada.
9. El uso de un mejor método de optimización, en este caso, algoritmos con estimación de distribuciones (EDA), mejora los resultados de Lesk completo en comparación con los de Lesk simple, a diferencia de lo antes reportado en el estado del arte.
10. Los resultados del método propuesto, Lesk simple con *back-off* a Lesk optimizado son muy similares a los que se obtienen para Lesk optimizado, ya que, al ser tan baja la cobertura de Lesk simple, Lesk optimizado decide la mayoría de los casos.

11. El método propuesto de Lesk simple modificado muestra mejores resultados en comparación con los de Lesk simple.
12. El método propuesto de Lesk simple modificado muestra mejores resultados en comparación con los de Lesk optimizado con EDA.

7.4 Aportaciones principales

7.4.1 Aportaciones teóricas

- Algoritmo para la desambiguación del sentido de las palabras, basado en la unión del algoritmo de Lesk simple y Lesk completo optimizado (llamado Lesk simple con *back-off* a Lesk optimizado).
- Algoritmo para la desambiguación del sentido de las palabras, basado en una modificación al algoritmo simplificado de Lesk (llamado Lesk simple modificado).
- Demostración de que un mejor método de optimización aumenta considerablemente los resultados para el método original de Lesk.
- Demostración de que, a diferencia de lo que se ha planteado por otros autores en el estado del arte, el método de Lesk completo con un mejor método de optimización muestra mejores resultados en comparación con los del método simplificado de Lesk.
- Análisis de la cobertura, precisión y *recall* obtenidos para los métodos para la desambiguación del sentido de las palabras basados en la aplicación directa de diccionario de sentidos, como son Lesk completo y Lesk simple; y la interacción que tienen estos algoritmos con diferentes estrategias de *back-off*.

7.4.2 Productos obtenidos

- Herramienta para desambiguación del sentido de las palabras, basado en la aplicación directa del diccionario, independiente del lenguaje.
- Datos experimentales de:
    1. Método de Lesk optimizado y Lesk simple evaluado en cuatro corpus diferentes y con diferentes medidas de similitud.

2. Interacción de los métodos tipo Lesk (optimizado y simple) con diferentes estrategias de *back-off*.
3. Evaluación de métodos tipo Lesk considerando diferente información en la definición del sentido de las palabras, es decir, considerando solamente la descripción de un sentido (glosa) o ejemplos o ambos.
4. Método original de Lesk (exhaustivo) con diferentes medidas de similitud.
5. Métodos base (*baseline)* sentido aleatorio y sentido más frecuente.

7.4.3 Trabajos publicados

- S. Torres and A. Gelbukh. *Comparing Similarity Measures for Original WSD Lesk Algorithm*. Research in Computing Science Vol. 43, ISSN: 1870-4069, pp. 155-166, 2009.
- M. Ríos Gaona, S. Torres and A. Gelbukh. *Evolutive Method for Word Sense Disambiguation.*(envíado a revista internacional)

7.5 Trabajo futuro

- Experimentar con ventanas de contexto de diferente tamaño para ver el comportamiento de los métodos tipo Lesk para desambiguación del sentido de las palabras.
- Usar otras medidas de similitud para el algoritmo simplificado de Lesk, dado que en este trabajo sólo se hicieron experimentos con diferentes medidas de similitud para el algoritmo de Lesk completo (Anexo 2).
- En lugar de *back-off* a sentido más frecuente, experimentar con el método de Lesk simple con *back-off* al método de McCarthy *et al*. (2004), ya que este método busca el sentido más frecuente de una palabra de una manera no supervisada.
- Analizar la influencia del diccionario en el desempeño de los métodos tipo Lesk para la desambiguación del sentido de las palabras.

Referencias

Referencias

(Agirre *et al.* 2007) Agirre E. and Edmons P. *Word Sense Disambiguation: Algorithms and Applications*. Springer. Text, Speech and Language Technology, Vol. 33. ISBN: 978-1-4020-6870-6, 2007

(Banerjee & Pedersen 2002) Banerjee and T. Pedersen. *An adapted Lesk algorithm for word sense disambiguation using WordNet*. In Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, February 2002.

(Bloomfield 1933) Bloomfield, Leonard. *Language.* Holt, New York, 1933.

(Bolshakov & Gelbukh 2001b) Bolshakov, I. A., A. Gelbukh. *Text segmentation to Paragraphs based on Local Text Cohesion.* In: V. Matousek *et al.* (Eds). *Text Speech and Dialogue.* Proc. 4th Intern. Conf. TSD-2001. Lecture Notes in Artificial Intelligence 2166, Springer, 2001, p. 158-166.

(Bolshakov & Gelbukh 2004) Bolshakov, Igor, Gelbukh, Alexander. *Computational Linguistics. Models, Resources, Applications.* Ciencia de la Computación. First Edition, México, 2004.

(Cowie 1992) Cowie, L., Guthrie, J. and Guthrie, L. *Lexical disambiguation using simulated annealing*. COLING 1992.

(Chomsky 1986) Chomsky, N. *Knowledge of language: Its nature, origin and use*. Praeger, New York, 1986.

(Dorigo *et al.* 1996) M. Dorigo, V. Maniezzo, A. Colorni, *The ant system: Optimization by a colony of cooperating agent*, IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol. 26, No. 1, 1996, pp. 29-41

(Franz 1996) Franz, A. *Automatic Ambiguity Resolution in Natural Language processing. An Empirical Approach.* Lecture Notes in Artificial Intelligence 1171. Springer Verlag Berlin Heidelberg, 1996

(Galicia-Haro *et al*. 1999) Galicia-Haro Sofía N., Bolshakov I. A. y Gelbukh A. F. *Un modelo de descripción de la estructura de las valencias de verbos españoles para el análisis automático de textos.*1999

(Galicia-Haro 2000) Galicia-Haro Sofía N., *Análisis sintáctico conducido por un diccionario de patrones de manejo sintáctico para lenguaje español.* Tesis doctoral, CIC, IPN, México, 2000.

(Galicia-Haro *et al*. 2001) Galicia-Haro Sofía N., Gelbukh A. F. y Bolshakov I. A. *Una aproximación para resolución de ambigüedad estructural empleando tres mecanismos diferentes.* J. Procesamiento de Lenguaje Natural, No 27, September 2001. SEPLN, Spain, 55-64, 2001.

(Gelbukh 2000) Gelbukh, Alexander. *Computational Processing of Natural Language: Tasks, Problems and Solutions.* Congreso Internacional de Computación en México D.F., Nov 15-17, 2000.

(Gelbukh *et al*. 2003) Alexander Gelbukh, Grigori Sidorov, Francisco Velásquez. 2003. *Análisis morfológico automático del español a través de generación. Escritos*, N 28, pp. 9 – 26.

(Gelbukh *et al*. 2003b) Alexander Gelbukh, Grigori Sidorov, Sang Yong Han. *Evolutionary Approach to Natural Language Word Sense Disambiguation through Global Coherence Optimization*. WSEAS Transactions on Communications, ISSN 1109-2742, Issue 1 Vol. 2, January 2003, p. 11–19.

(Glover 1986) Glover, F. *Future Paths for Integer Programming and Links to Artificial Intelligence.* Computers and Operations Research 13, 533-549, 1986

(Goldberg 1989) D.E. Goldberg, *Genetic algorithms in search, optimization, and machina learning*, Addison-Wesley, New York, 1989.

(Hirst 1987) Hirst, Graeme. *Semantic interpretation and the resolution of ambiguity.* Studies in Natural Language Processing. Cambridge University Press, Cambridge, United Kingdom, 263. 1987.

(Hirst 1998) Hirst, Graeme. *Chapter 13. Lexical Chains as Representations of Context for the Detection and Correction of Malapropisms*. WordNet An electronic Lexical Database. Edited by Christiane Fellbaum. The MIT Press. Cambridge, Massachusetts, London, England, 1998.

(Holland 1975) Holland, J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

(Ide & Veronis 1998) Ide N. and Veronis J. *Word Sense Disambiguation: The state of the art.* Computational Linguistics, 24 (1:1-40).

(Jiang & Conrath 1997) Jiang and D. Conrath. *Semantic similarity based on corpus statistics and lexical taxonomy*. In Proceedings on International Conference on Research in Computational Linguistics, Taiwan, 1997.

(Kennedy & Eberhart 1995) J. Kennedy, R.C. Eberhart, *Particle swarm optimization*, Proceedings of the IEEE International Conference on Neural Networks-ICNN'95, Perth (Australia), December 1995, Vol.4, pp. 1942-1948.

(Kilgarriff & Rosenzweig 2000) Kilgarriff, A. and Rosenzweig, J. *Framework and results for English SENSEVAL*. Computers and the Humanities, 2000, 34 (1-2).

(Kirkpatrick *et al*. 1983) S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, *Optimization by simulated annealing*, Science, Vol. 220, No. 4598, May 1983, pp. 671-680.

(Krovetz & Croft 1992) Krovetz, Robert y Croft, William Bruce. *Lexical Ambiguity and Information Retrieval*. ACM Transactions on Information Systems, 10(2), 115-141, 1992.

(Leacock *et al*. 1998) Leacock, C., Chodorow, M., and Miller, G., *Using corpus statistics and WordNet relations for sense identification.* Computational Linguistics, 24(1):147-165, 1998.

(Ledo-Mezquita 2005) Ledo-Mezquita, Yoel. *Recuperación de información con resolución de ambigüedad de sentidos de palabras para el español.* Tesis doctoral, CIC, IPN, México, 2005.

(Lesk 1986) Lesk, M., *Automatic sense disambiguation using machine-readable dictionaries: how to tell a pine cone from an ice cream cone*. Proc. of ACM SIGDOC Conference. Toronto, Canada, 1986, p. 24–26.

(Lin 1997) D. Lin. *Using syntactic dependency as a local context to resolve word sense ambiguity.* In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, pages 64–71, Madrid, July 1997.

(Lyons 1977) Lyons, J. *Semantics*. Cambridge, 1977.

(McCarthy *et al*. 2004) McCarthy, D., Koeling, R., Weeds, J. and Carroll, J. *Finding predominant senses in untagged text*. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain, 2004.

(Mihalcea 2005) Mihalcea, R. *Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling*. In Proceedings of HLT05, Morristown, NJ, USA. 2005

(Moscato 1989) P. Moscato, *On evolution, search, optimization, genetic algorithms and martial arts. Towards memetic algorithms*, Technical Report 158-79, Caltech Concurrent Computation Program, California Institute of Technology, 1989

(Penn 2004) Gerald Penn, *Word Sense Disambiguation*. CSC401, Spring 2004. University of Toronto

(Pollard & Sag 1987) Pollard, C. J. and I. A. Sag. *Information-based syntax and semantics*. CSLI Lecture notes series. Chicago University Press. Chicago II. Center for the Study of Language and Information; Lecture Notes Number 13, 1987.

(Resnik & Hearst 1993) Resnik, P. and Hearst, M. *Syntactic ambiguity and conceptual relations.* In: K. Church (ed.) Proceedings of the ACL Workshop on Very Large Corpora, 58-64, 1993.

(Resnik 1995) Resnik, P. *Using information content to evaluate semantic similarity in a taxonomy.* In Proceedings of the 14th International Joint Conference on Artificial Intelligence,Montreal, August 1995.

(Reynolds 1994) R.G. Reynolds, *An introduction to cultural algorithms*, Proceedings of the Third Annual Conference on Evolutionary Programming, World Scientific, River Edge, New Jersey, 1994, pp. 131-139.

(Salton 1968) Salton, Gerard. *Automatic Information organization and Retrieval*. McGraw-Hill, New Cork, 1968.

(Salton & McGill 1983) Salton, Gerard y McGill, M. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983

(Schütze & Pedersen 1995) Schütze, Hinrich y Pedersen, Jan. *Information retrieval based on word senses*. Proceedings of SDAIR'95. Las Vegas, Nevada, April, 1995.

(Sidorov 2001) Sidorov, G.. *Problemas actuales de lingüística computacional*. Revista digital universitaria, UNAM, México. Vol. 2 No. 1, 2001.

(Sidorov 2005a) Sidorov, G. *La capacidad lingüística de las computadoras*. Conversus, Vol. 36, 2005, pp. 28–37.

(Sidorov 2005b) Sidorov G. *Etiquetador Morfológico y Desambiguador Manual: Dos Aplicaciones del Analizador Morfológico Automático para el Español*. En: Memorias del VI encuentro internacional de computación ENC-2005, México, Puebla, 2005, pp. 147–149.

(Sinha & Mihalcea 2007) Ravi Sinha and Rada Mihalcea, *Unsupervised Graph-based Word Sense Disambiguation Using Measures of Word Semantic Similarity*, in Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2007), Irvine, CA, September 2007

(Tejada-Cárcamo 2006) Tejada Cárcamo Javier., *Desambiguación de sentidos de palabras usando relaciones sintácticas como contexto local.* Tesis de Maestría, CIC, IPN, México, 2006.

(Vasilescu *et al.* 2004) F. Vasilescu, P. Langlais, G. Lapalme "Evaluating variants of the Lesk approach for disambiguating words", LREC 2004.

(Vlach & Singhal 1983) J. Vlach and K. Singhal. Computer Methods for Circuit Analysis and Design Van Nostrand Reinhold Company NewYork, 1983, 584 pp.

(Voorhees 1993) Voorhees, Ellen M. *Using WordNet to disambiguate word senses for text retrieval*. Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 27 June-1 July 1993, Pittsburgh, Pennsylvania, 171-180, 1993.

(Weaver 1949) Weaver, Warren. *Translation. Mimeographed*, 12 pp., July 15, 1949. Reprinted in Locke, William N. y Booth, A. Donald (1955) (Eds.), Machine translation of languages. John Wiley and Sons, New York, 15-23. 1949.

(Wilks 1975) Wilks, Yorick A. *Preference semantics.* In Keenan, E. L. III (Ed.), Formal Semantics of Natural Language. Cambridge University Press, 329-348, 1975.

(Wilks *et al*. 1993) Wilks Y., Fass D., Guo C., McDonal J., Plate T. and Slator B. *Providing Machine Tractable dictionary tools*. In Semantics and the lexicon (Pustejowsky J. Ed.) 341-401, 1993.

(Wilks 1998) Wilks, Yorick A. *Senses and texts.* In Computers and the Humanities, 1998.

(Yarowsky 1993) Yarowsky, David. *One sense per collocation.* Proceeding of ARPA Human Language Technology Workshop, Princeton, New Jersey, 266-271, 1993.

(Yarowsky 1994) Yarowsky, David. *Decision lists for lexical ambiguity resolution: application to accent restoration in Spanish and French.* Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico, 88-95, 1994.

(Yarowsky 1995) Yarowsky, D. *Unsupervised word sense disambiguation rivaling supervised methods*. Proceedings of ACL 1995.

(Yngve 1955) Yngve, Victor H. Syntax and the problem of multiple meaning. In Locke, William N. and Booth, A. Donald (Eds.), Machine translation of languages. John Wiley & Sons, New York, 208-226, 1955

Anexos

Anexos

Anexo 1. Significado de las etiquetas utilizadas para la anotación del corpus Senseval, Semeval y Semcor.

Senseval, Semeval y Semcor son archivos de concordancia semántica. Una concordancia semántica consiste en textos que han sido etiquetados sintáctica y semánticamente.

El etiquetado semántico está hecho a mano, utilizando varias herramientas para anotar texto en inglés con sentidos de WordNet.

Formato

Independientemente de que los archivos comprenden una concordancia semántica, y cuáles palabras están etiquetas, el formato de cada archivo de concordancia es el mismo. El formato de un archivo de contexto sigue las líneas de SGML (***S**tandard **G**eneralized **M**arkup **L**anguage)*, usando elementos y pares de atributo/valor para registrar información acerca del archivo, fronteras de párrafo y oración, e información sintáctica y semántica.

Todos los elementos de SGML requieren tanto etiquetas de inicio como de finalización.

Los pares de atributo/valor de SGML siguen la forma:

atributo=valor

El formato de SGML de Semcor sólo se desvía del estándar en que el valor es encerrado entre comillas cuando puede contener espacios en blanco.

Debido al gran número de pares de atributo/valor, la presencia de comillas alrededor de cada *valor*, incrementa sustancialmente el número de concordancias.

Nomenclatura

La estructura de un archivo de contexto se especifica a continuación en pseudo-BNF notación. Cada elemento SGML se encuentra en una línea aparte. Las "terminales" se encuentran en **negritas** y están representadas en el archivo con están escritas. Los elementos en *cursiva* son variables. Las cadenas en MAYUSCULAS no son "terminales".

Estructura del archivo

CONTEXTFILE ::= **<contextfile concordance=** *conc* >
CONTEXT+
</contextfile>

CONTEXT ::= **<context filename=** *filename* **paras=yes>**
PARA+ | SENT+
</context>

PARA ::= **<p pnum=** *paragraph_number* >
SENT+
</p>

SENT ::= **<s snum=** *sentence_number* >
SENT_TOK+
</s>

SENT_TOK ::= (WORD_FORM | PUNC)+

WORD_FORM ::= **<wf cmd=tag** RDF SEP POS > *word* **</wf>**
| **<wf cmd=ignore** DC SEP POS > *word* **</wf>**
| **<wf cmd=done** RDF SEP POS SEM_TAG OT> *word* **</wf>**
| **<wf cmd=(update | retag)** RDF SEP POS TAGNOTE NOTE> *word* **</wf>**

POS ::= **pos=** POS_TAG

POS_TAG ::= **CC | CD | DT | EX | FW | IN | JJ | JJR | JJS | LS | MD | MD|VB**
| NN | NNP | NNPS| NNP|NP | NNP|VBN | NNS | NN|SYM | NP | NPS
| PDT | POS| PP | PR | PRP | PRP$ | RB | RBR | RBS | RP
| TO | UH | VB | VBD | VBG| VBN | VBP | VBZ | WDT | WP| WP$ | WRB

SEM_TAG ::= LEMMA WNSN LEXSN PN | NULL

LEMMA ::= **lemma=** *lemma*

WNSN ::= **wnsn=** *sense_number*

LEXSN ::= **lexsn=** *lex_sense*

PN ::= **pn=** CATEGORY | NULL

CATEGORY ::= **person** | **location** | **group** | **other**

RDF ::= **rdf=** *redefinition* | NULL

DC ::= **dc=** *distance* | NULL

SEP ::= **sep="** *separator_string* **"** | NULL

TAGNOTE ::= **tagnote=** TAGNOTE_TYPE

TAGNOTE_TYPE ::= **sns_miss** | **indist_sns** | **wd_miss** | **insuffctxt** | **sense_lost** | **misc**

NOTE ::= **note="** *note* **"**

OT ::= **ot=** OTHER_TAG | NULL

OTHER_TAG ::= **notag** | **metaphor** | **idiom** | **complexprep** | **foreignword** | **nonceword**

PUNC ::= **<punc>** PUNC_CHARACTER**</punc>**

PUNC_CHARACTER ::= [**, . ? ! , ; ([ )] ` ' $ " :**]

Interpretación de los elementos SGML

contextfile concordance= *conc* **>**

Este elemento indica el comienzo de un archivo de contexto. *conc* especifica el nombre de la concordancia semántica que se encuentra en el archivo. Un archivo de concordancia semántica contiene uno o más elementos de contexto de la misma concordancia semántica.

<context filename=*filename* paras=yes>

Este elemento indica el comienzo de un contexto. *filename* es el nombre del archivo del corpus original del cual se extrae el contexto. **paras** indica que este documento contiene delimitadores de párrafo.

<p pnum=*paragraph_number* >

Comienzo de un nuevo párrafo. *paragraph_number* es un entero. El primer párrafo en **contexto** es numerado **1**, y los números de párrafo son incrementados secuencialmente.

<s snum=*sentence_number* >

Inicio de una nueva oración. *sentence_number* es un entero. La primer oración en cada **contexto** es numerada **1**, y los números de oración son incrementados secuencialmente en todo el **contexto**. Los números de oración no reinician en uno en cada párrafo.

<wf *attribute/value_pairs* > *word* **</wf>**

Este elemento representa una palabra. *word* es la forma ortográfica tal y como aparece en el documento original. Toda la información sintáctica y semántica es almacenada en forma de pares de atributo/valor descrito abajo.

cmd= *cmd*

Indica el estatus de un elemento **wf.**

cmd	**Significado**
tag	Palabra que debe ser etiquetada
done	Palabra etiquetada semánticamente
ignore	Palabra que no debe ser etiquetada
update	Utilizada solo durante el desarrollo de concordancia semántica
retag	Utilizada solo durante el desarrollo de concordancia semántica

pos= *pos*

pos es la etiqueta sintáctica asignada por el etiquetador estocástico de categoría gramatical de Eric Brill. Ver *Etiquetas Sintácticas* abajo, para una lista de posibles valores.

lemma= *lemma*

La forma básica de la palabra o colocación que pertenece a los otros pares de atributo/valor en su **wf**. Esta es la forma de la cadena utilizada para buscar en la base de datos de WordNet. Si **rdf** está presente, *lemma* es la forma básica de la redefinición. Cuando **pn** está presente, *redefinition*, *lemma* y *category* tienen el mismo valor.

wnsn = *sense_number*

sense_number es el número de sentido (entero) correspondiente a la salida de pantalla de WordNet.

lexsn = *lex_sense*

lex_sense, cuando la encontramos concatenada con *lemma* usando el carácter de concatenación "**%"**, se crea una *sense_key* que indica a cual sentido de WordNet debemos ligar la palabra (*word)*. Esta es la etiqueta semántica de una palabra (*word).*

pn= *category*

Indica que la palabra (*word)* es un nombre propio clasificado como uno de los valores de CATEGORY. Cuando **pn** está presente, *redefinition* , *lemma* y *category* tienen el mismo valor.

rdf= *redefinition*

Si está presente, palabra(*word)* ha sido "redefinido" a algo más. Esto es principalmente usado para definir colocaciones discontinuas, corregir errores tipográficos en el texto, o entrar una cadena que debería usarse para buscar en WordNet en vez de la palabra (*word)* con el fin de encontrar un sentido apropiado para la etiqueta semántica. Cuando **pn** está presente, *redefinition* , *lemma* y *category* tienen el mismo valor.

dc= *distance*

Indica que una palabra(*word)* es parte de una colocación discontinua en la que las palabras que comprende la colocación no son adyacentes. *distance* es un entero que especifica cuántos elementos **wf** lejanos a la etiqueta semántica se encuentran en la colocación. Puede ser negativo, indicando cuántos elementos **wf** anteriores a éste, o positivo, indicando **wf** elementos siguientes en el archivo.

sep=" *separator_string* **"**

Indica que el espacio entre este elemento **wf** y el siguiente debe mostrarse como *separator_string* . La cadena puede ser de uno o más caracteres. El separador de palabra predeterminado es un espacio en blanco.

tagnote= *tagnote_type*

Un **tagnote** de pares de atributo/valor está siempre presente si *cmd* es **update** o **retag**. Éste es usado solo durante el desarrollo de la concordancia semántica, e indica el tipo de problema encontrado durante el etiquetado semántico.

note=" *note"*

Una **note** de pares de atributo/valor está siempre presente con **tagnote** . *note* puede contener una cadena que otorga información adicional sobre el **tagnote** , o puede estar vació.

ot= *other_tag*

Si está presente, una etiqueta semántica puede no ser asignada a la palabra (*word)* por una de las razones listadas en OTHER_TAG.

Etiquetas Sintácticas

Las siguientes son etiquetas asignadas por el etiquetador estocástico de categoría gramatical de Eric Brill.

Etiqueta sintáctica	Interpretación
CC	Conjunción coordinada
CD	Cardinalidad
DT	Determinante
EX	Existencial "there"
FW	Palabra extranjera
IN	Preposición o conjunción subordinada
JJ	Adjetivo
JJR	Adjetivo, comparativo
JJS	Adjetivo, superaltivo
LS	Marcador de elemento de lista
MD	Modal
NN	Sustantivo, singular o no contable
NNP	Nombre propio, singular
NNPS	Nombre propio, plural
NNS	Sustantivo, plural

NP	Nombre propio, singular
NPS	Nombre propio, plural
PDT	Predeterminante
POS	Terminación posesiva
PP	Pronombre personal
PR	Pronombre
PRP	Pronombre
PRP$	Pronombre, plural
RB	Adverbio
RBR	Adverbio, comparativo
RBS	Adverbio, superlativo
RP	Partícula
SYM	Símbolo
TO	"to"
UH	Intersección
VB	Verbo, forma básica
VBD	Verbo, tiempo pasado
VBG	Verbo, gerundio o presente participio
VBN	Verbo, pasado participio
VBP	Verbo, presente singular sin 3era persona
VBZ	Verbo, 3era persona presente singular
WDT	Wh-determinante
WP	Wh-pronombre
WP$	Wh-pronombre posesivo
WRB	Wh-adverbio

Ejemplos

En el primer ejemplo podemos ver una muestra del corpus para el primer párrafo (etiquetado <p pnum=1>) de un archivo, el cual contiene sólo una oración (etiquetada <s snum=1>). Al principio se encuentra la información del archivo de contexto como es: nombre del archivo y delimitadores de párrafo.

La oración es: *Committee approval of Gov._Price_Daniel 's "abandoned property" act seemed certain Thursday despite the adamant protests of Texas bankers.*

```
<contextfile concordance=brown>
<context filename=br-a02 paras=yes>
<p pnum=1>
<s snum=1>
<wf cmd=done pos=NN lemma=committee wnsn=1 lexsn=1:14:00::>Committee</wf>
<wf cmd=done pos=NN lemma=approval wnsn=1 lexsn=1:04:02::>approval</wf>
<wf cmd=ignore pos=IN>of</wf>
<wf cmd=done rdf=person pos=NNP lemma=person wnsn=1 lexsn=1:03:00::
    pn=person>Gov._Price_Daniel</wf>
<wf cmd=ignore pos=POS>'s</wf>
<punc>``</punc>
<wf cmd=done pos=JJ lemma=abandoned wnsn=1
lexsn=5:00:00:uninhabited:00>abandoned</wf>
<wf cmd=done pos=NN lemma=property wnsn=2 lexsn=1:21:00::>property</wf>
<punc>"</punc>
<wf cmd=done pos=NN lemma=act wnsn=1 lexsn=1:10:01::>act</wf>
<wf cmd=done pos=VB lemma=seem wnsn=1 lexsn=2:39:00::>seemed</wf>
<wf cmd=done pos=JJ lemma=certain wnsn=4 lexsn=3:00:03::>certain</wf>
<wf cmd=done pos=NN lemma=thursday wnsn=1 lexsn=1:28:00::>Thursday</wf>
<wf cmd=ignore pos=IN>despite</wf>
<wf cmd=ignore pos=DT>the</wf>
<wf cmd=done pos=JJ lemma=adamant wnsn=1 lexsn=5:00:00:inflexible:02>adamant</wf>
<wf cmd=done pos=NN lemma=protest wnsn=1 lexsn=1:10:00::>protests</wf>
<wf cmd=ignore pos=IN>of</wf>
<wf cmd=done pos=NN lemma=texas wnsn=1 lexsn=1:15:00::>Texas</wf>
<wf cmd=done pos=NN lemma=banker wnsn=1 lexsn=1:18:00::>bankers</wf>
<punc>.</punc>
</s>
</p>
```

En este ejemplo podemos ver una muestra del corpus para un párrafo que contiene varias oraciones. Al principio se encuentra la información del archivo de contexto como es: nombre del archivo y delimitadores de párrafo.

El párrafo es: "*I had a rather small place of my own. A nice bachelor apartment in a place called the Lancaster_Arms".*

```
<contextfile concordance=brown1>
<context filename=br-p12 paras=yes>
<p pnum=1>
<s snum=1>
<punc>``</punc>
```

<wf cmd=done pos=PRP ot=notag>I</wf>
<wf cmd=done pos=VB lemma=have wnsn=4 lexsn=2:40:04::>had</wf>
<wf cmd=ignore pos=DT>a</wf>
<wf cmd=done pos=RB lemma=rather wnsn=2 lexsn=4:02:04::>rather</wf>
<wf cmd=done pos=JJ lemma=small wnsn=1 lexsn=3:00:00::>small</wf>
<wf cmd=done pos=NN lemma=place wnsn=7 lexsn=1:15:06::>place</wf>
<wf cmd=ignore pos=IN>of</wf>
<wf cmd=ignore pos=PRP$>my</wf>
<wf cmd=done pos=JJ lemma=own wnsn=1 lexsn=5:00:00:personal:00>own</wf>
<punc>.</punc>
</s>
<s snum=2>
<wf cmd=ignore pos=DT>A</wf>
<wf cmd=done pos=JJ lemma=nice wnsn=1 lexsn=3:00:00::>nice</wf>
<wf cmd=done pos=NN lemma=bachelor wnsn=1 lexsn=1:18:00::>bachelor</wf>
<wf cmd=done pos=NN lemma=apartment wnsn=1 lexsn=1:06:00::>apartment</wf>
<wf cmd=ignore pos=IN>in</wf>
<wf cmd=ignore pos=DT>a</wf>
<wf cmd=done pos=NN lemma=place wnsn=2 lexsn=1:15:04::>place</wf>
<wf cmd=done pos=JJ lemma=called wnsn=1 lexsn=5:00:00:titled:00>called</wf>
<wf cmd=ignore pos=DT>the</wf>
<wf cmd=done rdf=location pos=NNP lemma=location wnsn=1 lexsn=1:03:00::
pn=location>Lancaster_Arms</wf>
<punc>"</punc>
<punc>.</punc>
</s>

En este ejemplo podemos ver una muestra del corpus para un párrafo que contiene dos oraciones, en las que sólo se han etiquetado semánticamente los verbos.

El párrafo es: "*We 'll grab horses", Dean said. "The main bunch is, outside but there are some over there inside the wall".*

<s snum=20>
<punc>``</punc>
<wf cmd=ignore pos=PRP>We</wf>
<wf cmd=ignore pos=MD>'ll</wf>
<wf cmd=done pos=VB lemma=grab wnsn=1 lexsn=2:35:00::>grab</wf>
<wf cmd=tag pos=NNS>horses</wf>
<punc>"</punc>
<punc>,</punc>
<wf cmd=tag pos=NNP>Dean</wf>
<wf cmd=done pos=VB lemma=say wnsn=1 lexsn=2:32:00::>said</wf>
<punc>.</punc>
</s>

<s snum=21>
<punc>``</punc>
<wf cmd=ignore pos=DT>The</wf>
<wf cmd=tag pos=JJ>main</wf>
<wf cmd=tag pos=NN>bunch</wf>
<wf cmd=done pos=VB lemma=be wnsn=3 lexsn=2:42:05::>is</wf>
<wf cmd=tag pos=RB>outside</wf>
<punc>,</punc>
<wf cmd=ignore pos=CC>but</wf>
<wf cmd=ignore pos=EX>there</wf>
<wf cmd=done pos=VB lemma=be wnsn=5 lexsn=2:42:04::>are</wf>
<wf cmd=ignore pos=DT>some</wf>
<wf cmd=ignore pos=IN>over</wf>
<wf cmd=tag pos=RB>there</wf>
<wf cmd=ignore pos=IN>inside</wf>
<wf cmd=ignore pos=DT>the</wf>
<wf cmd=tag pos=NN>wall</wf>
<punc>''</punc>
<punc>.</punc>
</s>
</p>

Anexo 2. Resultados obtenidos para el método de Lesk completo y Lesk simple evaluados en cuatro corpus y con diferentes medidas de similitud semántica

Los resultados que se presentan a continuación están dados por corpus. Cada corpus fue evaluado con varios algoritmos de desambiguación como: Lesk simple, Lesk completo (exhaustivo), y Lesk optimizado usando EDA. El algoritmo simplificado de Lesk sólo fue evaluado con traslape como medida de similitud.

Para obtener los resultados de Lesk completo (exhaustivo) se necesita mucho tiempo, debido al gran número de combinaciones de sentidos para cada oración. Por ello se muestran los resultados en dos bloques: "Con oraciones largas" y "Sin oraciones largas", en este último bloque eliminamos las oraciones con más de 210,567,168,000 de combinaciones.

Senseval 2

Datos depurados

- El bloque "Con oraciones largas":

235 oraciones, se eliminaron 3.

2362 palabras, se eliminaron 74.

De las 2362 palabras 448 son adjetivos, 286 son adverbios, 1111 son sustantivos y 517 son verbos. En la tabla siguiente se puede ver la distribución de estos datos (de acuerdo su categoría gramatical) para cada archivo de Senseval 2.

Tabla 21. Datos de la categoría gramatical de las palabras (depuradas) para Senseval 2

	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
ARCHIVO 1	99	86	331	162
ARCHIVO 2	170	107	495	242
ARCHIVO 3	188	106	310	140

- El bloque "Sin oraciones largas":

220 oraciones, se eliminaron 18, de las cuales 15 eran largas.

2087 palabras, se eliminaron 349 de las cuales 275 corresponden a las de las oraciones largas removidas.

De las 2087 palabras desambiguadas 387 son adjetivos, 252 son adverbios, 992 son sustantivos y 456 son verbos. En la tabla 22 se puede ver la distribución de estos datos (de acuerdo su categoría gramatical) para cada archivo de Senseval 2.

Tabla 22. Datos de la categoría gramatical de las palabras (depuradas) para Senseval 2 (sin oraciones largas)

	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
ARCHIVO 1	73	64	272	132
ARCHIVO 2	152	98	451	210
ARCHIVO 3	162	90	269	114

Lesk simple

– *Con back-off a sentido más frecuente*

- Con oraciones largas:

Archivo	D00	D01	D02
Total de palabras	667	986	709
Errores	284	365	277
% de precisión	57.4	63	60.9
Total	60.8		

- Sin oraciones largas:

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	218	329	244
% de precisión	59.7	63.9	61.6
Total	62.1		

– *Con back-off a sentido aleatorio*

- Con oraciones largas:

Archivo	D00	D01	D02
Total de palabras	667	986	709
Errores	386	517	376
% de precisión	42.1	47.6	47
Total	45.9		

- Sin oraciones largas:

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	309	477	335
% de precisión	42.9	47.6	47.2
Total	46.3		

Lesk completo exhaustivo

Resultados obtenidos para el método de Lesk completo exhaustivo, sin oraciones largas, con back-off a sentido más frecuente:

- Medida de similitud: JCN

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	223	350	251
% de precisión	58.8	61.6	60.5
Total	60.5		

- Medida de similitud: LESK ADAPTADA

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	263	374	309
% de precisión	51.4	59	51.3
Total	54.7		

- Medida de similitud: COMBINACIÓN DE MEDIDAS

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	262	357	296
% de precisión	51.6	60.8	53.4
Total	56.2		

Resultados obtenidos para el método de Lesk completo exhaustivo, sin oraciones largas, con back-off a sentido aleatorio:

- Medida de similitud: JCN

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	297	449	290
% de precisión	45.1	50.7	54.3
Total	50.4		

- Medida de similitud: LESK ADAPTADA

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	264	374	309
% de precisión	51.2	59	51.3
Total	54.6		

- Medida de similitud: COMBINACIÓN DE MEDIDAS

Archivo	D00	D01	D02
Total de palabras	541	911	635
Errores	263	381	296
% de precisión	51.4	58.2	53.4
Total	55		

Lesk optimizado usando EDA

Resultados para el método de Lesk completo utilizando el algoritmo de estimación de distribuciones, con back-off sentido aleatorio:

- Con oraciones largas:

Medida usada	JCN			LESK			LCH			COMBINADO		
Archivo	D00	D01	D02	D00	D01	D02	D00	D01	D02	D00	D01	D02
Palabras total	667	986	709	667	986	709	667	986	709	667	986	709
Errores	329	409	332	333	429	346	396	531	370	313	402	348
% de precisión	50.7	58.5	53.2	50.1	56.5	51.2	40.6	46.1	47.8	53.1	59.2	50.9
% total	54.7			53.1			45.1			55		

Medida usada	LIN			PATH			RES			VECTOR		
Archivo	D00	D01	D02	D00	D01	D02	D00	D01	D02	D00	D01	D02
Palabras total	667	986	709	667	986	709	667	986	709	667	986	709
Errores	367	504	379	393	522	370	388	552	411	390	502	353
% de precisión	45	48.9	46.5	41.1	47.1	47.8	41.8	44	42	41.5	49.1	50.2
% total	47.1			45.6			42.8			47.3		

- Sin oraciones largas:

Medida usada	JCN			LESK ADAPTADA			LCH			COMBINADO		
Archivo	D00	D01	D02	D00	D01	D02	D00	D01	D02	D00	D01	D02
Palabras total	541	911	635	541	911	635	541	911	635	541	911	635
Errores	255	378	284	267	396	301	313	486	322	252	373	305
% de precisión	52.9	58.5	55.3	50.6	56.5	52.6	42.1	46.7	49.3	53.4	59.1	52
% total	56.1			53.8			46.3			55.9		

Senseval 3

Datos Depurados

- El bloque "Con oraciones largas":

262 oraciones, se eliminaron 38.

2061 palabras, se eliminaron 20.

De las 2061 palabras 360 son adjetivos, 14 son adverbios, 950 son sustantivos y 737 son verbos. En la tabla 23 se puede ver la distribución de estos datos (de acuerdo su categoría gramatical) para cada archivo de Senseval 3.

Tabla 23. Datos de la categoría gramatical de las palabras (depuradas) para Senseval 3

	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
ARCHIVO 1	101	11	326	352
ARCHIVO 2	174	1	339	153
ARCHIVO 3	85	2	285	232

- El bloque "Sin oraciones largas":

238 oraciones, se eliminaron 62 de las cuales 24 eran largas.

1563 palabras, se eliminaron 518 de las cuales 498 corresponden a las de las oraciones largas removidas.

De las 1563 palabras 238 son adjetivos, 12 son adverbios, 716 son sustantivos y 597 son verbos. En la tabla 24 se puede ver la distribución de estos datos de acuerdo a cada archivo de Senseval 3.

Tabla 24. Datos de la categoría gramatical de las palabras (depuradas) para Senseval 3 (sin oraciones largas)

	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
ARCHIVO 1	85	10	286	320
ARCHIVO 2	79	1	175	82
ARCHIVO 3	74	1	255	195

Lesk simple

– *Con back-off a sentido más frecuente*

- Con oraciones largas:

Archivo	D000	D001	D002

Total de palabras	790	667	604
Errores	394	315	278
% de precisión	50.1	52.8	54
Total	52.1		

- Sin oraciones largas:

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	337	153	237
% de precisión	51.9	54.6	54.9
Total	53.5		

– *Con back-off a sentido aleatorio*

- Con oraciones largas:

Archivo	D000	D001	D002
Total de palabras	790	667	604
Errores	505	374	345
% de precisión	36.1	43.9	42.9
Total	40.6		

- Sin oraciones largas:

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	445	192	303
% de precisión	36.5	43	42.3
Total	39.9		

Lesk completo exhaustivo

Resultados obtenidos para Lesk completo exhaustivo, sin oraciones largas, back-off a sentido más frecuente:

- Medida de similitud: JCN

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	355	157	223
% de precisión	49.4	53.4	57.5
Total	53		

- Medida de similitud: LESK ADAPTADA

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	381	159	245

% de precisión	45.6	52.8	53.3
Total	49.8		

- Medida de similitud: COMBINACIÓN DE MEDIDAS

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	337	157	230
% de precisión	51.9	53.4	56.2
Total	53.7		

Resultados obtenidos por Lesk completo exhaustivo, sin oraciones largas, back-off a sentido aleatorio:

- Medida de similitud: JCN

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	401	181	274
% de precisión	42.8	46.3	47.8
Total	45.2		

- Medida de similitud: LESK ADAPTADA

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	382	161	244
% de precisión	45.5	52.2	53.5
Total	49.6		

- Medida de similitud: COMBINACIÓN DE MEDIDAS

Archivo	D000	D001	D002
Total de palabras	701	337	525
Errores	397	170	240
% de precisión	43.4	50	54.3
Total	48.4		

Lesk optimizado usando EDA

Resultados utilizando el algoritmo de estimación de distribuciones, back-off a sentido aleatorio:

- Con oraciones largas:

Medida usada	JCN			LESK ADAPTADA			LCH			COMBINADO		
Archivo	D000	D001	D002	D000	D001	D002	D000	D001	D002	D000	D001	D002
Palabras total	790	667	604	790	667	604	790	667	604	790	667	604
Errores	407	343	266	426	336	272	490	399	332	395	334	266

% de precisión	48.5	48.6	56	46.1	49.6	55	38	40.2	45	50	49.9	56
% total	50.7			49.8			40.8			51.7		

- Sin oraciones largas:

Medida usada	JCN			LESK ADAPTADA			LCH			COMBINADO		
Archivo	D000	D001	D002	D000	D001	D002	D000	D001	D002	D000	D001	D002
Palabras total	701	337	525	701	337	525	701	337	525	701	337	525
Errores	359	163	226	383	161	238	431	192	287	348	157	234
% de precisión	48.8	51.6	57	45.4	52.2	54.7	38.5	43	45.3	50.4	53.4	55.4
% total	52.1			50			41.8			52.7		

Semeval

Datos depurados

En este corpus no se eliminaron oraciones para Lesk completo exhaustivo, ya que todas las oraciones eran cortas (la oración mas larga tiene 8,263,775,520 combinaciones de sentidos). Los datos depurados son:

101 oraciones, se eliminaron 25.

476 palabras, se eliminaron 14.

De las 476 palabras 163 son sustantivos y 313 son verbos. En la tabla siguientes se puede ver la distribución de estos datos (de acuerdo su categoría gramatical) para cada archivo de Semeval.

Tabla 25. Datos de la categoría gramatical de las palabras (depuradas) para Semeval

	SUSTANTIVOS	VERBOS
ARCHIVO 1	44	73
ARCHIVO 2	60	98
ARCHIVO 3	59	142

Lesk simple

– *Con back-off a sentido más frecuente*

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	61	73	88
% de precisión	52.1	46.2	43.8

Total	46.6

– *Con back-off a sentido aleatorio*

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	29	39	51
% de precisión	24.8	24.7	25.4
Total	25		

Lesk completo exhaustivo

Resultados obtenidos para Lesk completo exhaustivo, con back-off a sentido más frecuente:

- Medida de similitud: JCN

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	46	58	64
% de precisión	39.3	36.7	31.8
Total	35.3		

- Medida de similitud: LESK ADAPTADA

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	44	58	72
% de precisión	37.6	36.7	35.8
Total	36.6		

- Medida de similitud: COMBINACIÓN DE MEDIDAS

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	41	61	76
% de precisión	35	38.6	37.8
Total	37.4		

Resultados obtenidos para Lesk completo exhaustivo, con back-off a sentido aleatorio:

- Medida de similitud: JCN

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	43	52	60
% de precisión	36.8	32.9	29.9
Total	32.6		

- Medida de similitud: LESK ADAPTADA

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	43	58	72
% de precisión	36.8	36.7	35.8
Total	36.3		

- Medida de similitud: COMBINACIÓN DE MEDIDAS

Archivo	D00	D01	D02
Total de palabras	117	158	201
Correctas	40	61	72
% de precisión	34.2	38.6	35.8
Total	36.3		

Lesk optimizado usando EDA

Resultados utilizando el algoritmo de estimación de distribuciones, back-off a sentido aleatorio:

Medida usada	JCN			LESK ADAPTADA			LCH			COMBINADO		
Archivo	D00	D01	D02	D00	D01	D02	D00	D01	D02	D00	D01	D02
Palabras total	117	158	201	117	158	201	117	158	201	117	158	201
Correctas	45	57	66	46	58	78	42	42	59	47	57	81
% de precisión	38.5	36.1	32.8	39.3	36.7	38.8	35.9	26.6	29.4	40.2	36.1	40.3
% total	35.3			38.2			30			38.9		

Semcor 2.1

Muestra

Para los experimentos con SemCor, se tomaron aleatoriamente 21 archivos de todo el corpus, ya que éste contiene muchas oraciones y se llevaría mucho tiempo procesarlas todas.

Las pruebas se hicieron sobre la siguiente muestra:

- un archivo llamado br-n15 que contiene 176 oraciones, 1035 palabras.
- 7 archivos de carpeta1, que contienen 664 oraciones, 7009 palabras.
- 6 archivos de carpeta2, que contienen 719 oraciones, 4870 palabras.
- 7 archivos de carpeta3, que contienen 1133 oraciones, 2376 palabras.

Dando un total de, 2692 oraciones, 15290 palabras.

En la tabla siguiente se puede ver la distribución de estos datos (de acuerdo a su categoría gramatical) para cada la muestra descrita anteriormente.

Tabla 26. Datos de la categoría gramatical de las palabras para la muestra de Semcor 2.1

MUESTRA	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
br-n15	146	113	463	313
7 archivos de la carpeta 1	1094	457	3875	1583
6 archivos de la carpeta 2	676	598	1933	1663
7 archivos de la carpeta 3	-	-	-	2376
Total	*1916*	*1168*	*6271*	*5935*

Datos depurados

- El bloque "Con oraciones largas":

2169 oraciones, se eliminaron 523.

14663 palabras, se eliminaron 627.

De las 14663 palabras 1886 son adjetivos, 1064 son adverbios, 6234 son sustantivos y 5479 son verbos. En la tabla 27 se puede ver la distribución de estos datos (de acuerdo a su categoría gramatical) para cada archivo de la muestra de Semcor 2.1.

Tabla 27. Datos de la categoría gramatical de las palabras (depuradas) para la muestra de Semcor 2.1

MUESTRA	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
br-n15	144	106	458	308
7 archivos de la carpeta 1	1082	437	3865	1579
6 archivos de la carpeta 2	660	521	1911	1610
7 archivos de la carpeta 3	-	-	-	1982

- El bloque "Sin oraciones largas":

2053 oraciones, se eliminaron 639 de las cuales 116 son oraciones largas.

12300 palabras, se eliminaron 2990 de las cuales 2363 corresponden a las de las oraciones largas.

De las 12300 palabras 1484 son adjetivos, 877 son adverbios, 5082 son sustantivos y 4857 son verbos. En la tabla siguiente se puede ver la distribución de estos datos (de acuerdo a su categoría gramatical) para cada archivo de la muestra de Semcor 2.1.

Tabla 28. Datos de la categoría gramatical de las palabras (depuradas) para la muestra de Semcor 2.1 (sin oraciones largas)

MUESTRA	ADJETIVOS	ADVERBIOS	SUSTANTIVOS	VERBOS
br-n15	136	103	434	295
7 archivos de la carpeta 1	861	362	3148	1319
6 archivos de la carpeta 2	487	412	1500	1261
7 archivos de la carpeta 3	-	-	-	1982

Lesk simple

– *Con back-off a sentido más frecuente*

- Con oraciones largas:

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	1016	6963	4702	1982
Errores	396	2436	1865	898
% de precisión	61	65	60.3	54.7
% total	61.8			

- Sin oraciones largas:

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	374	1902	1360	898
% de precisión	61.4	66.6	62.8	54.7
% total	63.1			

– *Con back-off a sentido aleatorio*

- Con oraciones largas:

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	1016	6963	4702	1982
Errores	592	3594	2601	1483
% de precisión	41.7	48.4	44.7	25.2
% total	43.6			

- Sin oraciones largas:

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	564	2930	2032	1483
% de precisión	41.7	48.5	44.5	25.2
% total	43			

Lesk completo exhaustivo

Resultados obtenidos para Lesk completo exhaustivo, sin oraciones largas, con back-off a sentido más frecuente:

- Medida de similitud: JCN

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	364	1665	1306	1108
% de precisión	62.4	70.7	64.3	44.1
% total	63.9			

- Medida de similitud: LESK ADAPTADA

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	435	2223	1587	1233
% de precisión	55.1	60.9	56.6	37.8
% total	55.5			

- Medida de similitud: COMNBINACIÓN DE MEDIDAS

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	397	1810	1361	1154
% de precisión	59	68.2	62.8	41.8
% total	61.6			

Resultados obtenidos para Lesk completo exhaustivo, sin oraciones largas, con back-off a sentido aleatorio:

- Medida de similitud: JCN

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	515	2244	1873	1131

% de precisión	46.8	60.6	48.8	42.9
% total	53.1			

- Medida de similitud: LESK ADAPTADA

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	441	2227	1590	1241
% de precisión	54.5	60.9	56.6	37.4
% total	55.3			

- Medida de similitud: COMBINACIÓN DE MEDIDAS

Muestra	br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Total de palabras	968	5690	3660	1982
Errores	467	2417	1910	1552
% de precisión	51.8	57.5	47.8	21.7
% total	48.4			

Lesk optimizado usando EDA

Resultados utilizando el algoritmo de estimación de distribuciones, back-off a sentido aleatorio:

- Con oraciones largas:

Medida usada	JCN			
Muestra	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	1016	6963	4702	1982
Errores	442	2329	1982	1108
% precisión	56.5	66.6	57.8	44.1
%total	60			

Medida usada	LESK ADAPTADA			
Muestra	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	1016	6963	4702	1982
Errores	478	2888	2080	1211
% precisión	53	58.5	55.8	38.9
%total	54.6			

Medida usada	LCH			
Muestra	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	1016	6963	4702	1982

Errores	571	3476	2558	1334
% precisión	43.8	50.1	45.6	32.7
%total	45.9			

Medida usada	COMBINADO			
Muestra	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	1016	6963	4702	1982
Errores	438	2590	2035	1323
% precisión	56.9	62.8	56.7	33.2
%total	56.4			

- Sin oraciones largas:

Medida usada	JCN			
Datos	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	968	5690	3660	1982
Errores	416	1866	1520	1110
% precisión	57	67.2	58.5	44
%total	60.1			

Medida usada	LESK ADAPTADA			
Datos	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	968	5690	3660	1982
Errores	449	2321	1595	1211
% precisión	53.6	59.2	56.4	38.9
%total	54.7			

Medida usada	LCH			
Datos	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	968	5690	3660	1982
Errores	538	2789	1941	1334
% precisión	44.4	51	47	32.7
%total	46.3			

Medida usada	COMBINADO			
Datos	un archivo br-n15	7 archivos de carpeta1	6 archivos de carpeta2	7 archivos de carpeta3
Palabras total	968	5690	3660	1982
Errores	413	2095	1568	1323
% precisión	57.3	63.2	57.2	33.2
%total	56.1			

Glosario

Glosario

Acento diacrítico	Se denomina *acento diacrítico* a la tilde (acento gráfico) que se emplea en palabras, habitualmente monosílabas, para diferenciar distintos significados para una misma palabra.
Ambigüedad	Término que hace referencia a aquellas estructuras gramaticales que pueden entenderse de varios modos o admitir distintas interpretaciones y dar, por consiguiente, motivo a dudas, incertidumbre o confusión.
Ambigüedad léxica	La ambigüedad léxica es aquella que se presenta en la categoría gramatical de un vocablo.
Ambigüedad semántica	La ambigüedad semántica es aquella que se presenta en una expresión, de tal manera que ésta puede expresar diferentes sentidos dependiendo del contexto local, el tópico global y el mundo pragmático en el que se manifiesta.
Ambigüedad sintáctica	La ambigüedad sintáctica, también conocida como estructural, es aquella que se presenta en oraciones, de tal manera que éstas puedan ser representadas por más de una estructura sintáctica.
Analizador sintáctico	Un analizador sintáctico para un lenguaje natural, es un programa que construye árboles de estructura de frase o de derivación para las oraciones de dicho lenguaje, además de proporcionar un análisis gramatical, separando los términos en constituyentes y etiquetando cada uno de ellos. Asimismo, puede proporcionar información adicional acerca de las clases semánticas (persona, género) de cada palabra y también la clase funcional (sujeto, objeto directo, etc.) de los constituyentes de la oración.

Aprendizaje no supervisado

Aprendizaje no basado en ejemplos. En aprendizaje automático no supervisado, el algoritmo descubre las reglas o estructuras analizando datos que se le presentan sin etiquetado manual. Usualmente se efectúe con algún tipo de optimización.

Aprendizaje supervisado

Aprendizaje basado en ejemplos. En el caso de la desambiguación supervisada se entrena un clasificador usando un corpus de texto etiquetado a mano.

Categoría gramatical

El término categoría gramatical o parte de la oración, que en inglés se denomina POS (*part-of-speech*) es una clasificación de las palabras de acuerdo a la función que desempeñan en la oración. La gramática tradicional distingue nueve categorías gramaticales: sustantivo, determinante, adjetivo, pronombre, preposición, conjunción, verbo, adverbio, interjección. No obstante, para algunos lingüistas, las categorías gramaticales son una forma de clasificar ciertos rasgos gramaticales, como por ejemplo: modo, aspecto, tiempo y voz.

Coherencia global

Al hablar de WSD, la coherencia global se refiere a que todos los sentidos de todas las palabras en cierta *ventana de contexto*, están co-relacionados, de forma que el sentido de una palabra está coordinado con el sentido de las demás.

Coherencia local

Al hablar de WSD, la coherencia local se refiere a que el sentido de una palabra no está coordinado con los sentidos de las demás palabras de su contexto, es decir, el sentido de esa palabra se puede decidir sin tomar en cuenta el sentido de las demás.

Colocación gramatical

Una colocación gramatical es un conjunto de dos o más palabras las cuales expresan una idea específica. El significado que expresa cada término de una colocación difiere de la semántica que dichos términos proporcionan cuando se usan de manera conjunta.

Conocimiento Representación simbólica de ideas, conceptos, nociones, hechos, seres, acciones, y de las relaciones entre ese tipo de elementos que reflejan un dominio del universo físico o del mundo de las ideas.

Desambiguación Eliminación de ambigüedades.

Diccionario computacional Un diccionario computacional surge al convertir un diccionario normal, creado exclusivamente para el uso humano, a formato electrónico. Estos diccionarios proveen información sobre sentidos de vocablos ambiguos, lo cual puede ser explotado por el área de desambiguación de sentidos de palabras.

Dominio El término dominio hace referencia a la temática general que expresa un documento o texto en su totalidad.

Etiqueta semántica Este término se utiliza para hacer referencia a un sentido específico de un vocablo ambiguo, tomando en cuenta alguna fuente de información. Por ejemplo, WordNet.

Etiqueta sintáctica Se refiere a la combinación de letras y números que se agrega como información a una palabra para identificar su categoría gramatical.

Etiquetado Lingüístico Se refiere a etiquetado de corpus, ya sea sintáctico, semántico, etc.

Fonología La fonología es el estudio de los sonidos del lenguaje. La fonética es la parte de la fonología que trata de la manera en que se pronuncian los sonidos y su forma acústica, pero la fonología también incluye el estudio de la manera en que los sonidos funcionan sistemáticamente en la lengua. Las otras divisiones importantes de ese estudio incluyen el análisis de los fonemas, los cambios de un sonido a otro en ciertos contextos (la alternancia morfofonémica y alofónica), la estructura de las sílabas, el acento, la entonación, y el tono lingüístico.

Frase Grupo de una o más palabras que funciona como unidad, pero que (normalmente) no funciona en su totalidad independientemente, como en el caso de una oración. A veces se marcan las frases, como las oraciones, poniéndolas entre corchetes. Por ejemplo: [las montañas [más altas]] es una frase nominal, y también contiene una frase adjetival, [más altas]. Contrástese con oración.

Gramática Es la manera característica en que se combinan los elementos básicos (especialmente los elementos léxicos) de una lengua, para formar estructuras más complejas que permitan la comunicación de los pensamientos. La gramática incluye la morfología y la sintaxis; algunos analistas incluyen la fonología, la semántica y el léxico también como parte de la gramática.

Herramientas lingüísticas Esta expresión hace referencia a diversos programas o aplicaciones usados en el procesamiento de lenguaje natural, tales como analizadores sintácticos, morfológicos, corpus, diccionarios electrónicos, ontologías, entre otros.

Hiperónimo Un hiperónimo es una palabra cuyo significado incluye al de otra u otras. Por ejemplo, pájaro respecto a jilguero y gorrión

Hipónimo Un homónimo es una palabra cuyo significado está incluido en el de otra. Por ejemplo, gorrión respecto a pájaro.

Lema, Lematización Término que en latín significa “forma canónica”. En lexicografía, entrada léxica del diccionario en que se suministra diversa información y es a menudo representativa de distintas formas flexionadas; p. ej. “ir” es el lema de “voy”, “vas”, “íbamos”, “fueron” y el resto de sus formas conjugadas. En cuanto a lematización, en lexicografía se denomina ‘lematización’ el proceso de reducción de las diferentes formas flexivas de una palabra a la forma canónica que

se selecciona como lema.

Lenguaje natural Es un término ya adoptado que el lenguaje humano se denomine natural para diferenciarlo de los lenguajes artificiales en el área de la computación.

Lexema Unidad léxica abstracta que no puede descomponerse en otras menores, aunque sí combinarse con otras para formar compuestos, y que posee un significado definible por el diccionario, no por la gramática. Por ejemplo: fácil es el lexema básico de facilidad, facilitar, fácilmente.

Léxico El conjunto de los morfemas de una lengua, junto con raíces complejas o palabras pre-formadas (o sea que no se arman en forma productiva), modismos y otras frases establecidas. Éstas son las estructuras lingüísticas que un hablante sabe como unidades completas y que puede usar sin tener que determinar sus significados a base de sus partes integrantes. Un diccionario (que a veces también se llama léxico) es un libro que exhibe elementos del léxico de una lengua, especialmente palabras, con una indicación breve de sus significados y usos. Contrástese con gramática; sintaxis, morfología, fonología, semántica.

Lingüística computacional La lingüística computacional puede considerarse una disciplina de la lingüística aplicada y la inteligencia artificial. Tiene como objetivo la creación e implementación de programas computacionales que permitan la comunicación entre el hombre y la computadora, ya sea mediante texto o voz.

Malapropismo Fenómeno lingüístico que consiste en sustituir una palabra por otra que tiene un sonido parecido, pero su significado es diferente.

Metasintaxis Sintaxis que se utiliza para definir otras sintaxis.

Objeto La gramática tradicional proporcionó los términos transitividad y objeto (tema de la siguiente sección), por el momento consideramos solamente la definición en el Diccionario de la Real Academia de la lengua Española: los transitivos son los verbos cuya acción recae en la persona o cosa que es término o complemento de la acción. De lo cual se define el complemento directo (objeto) como el complemento en el cuál recae directamente la acción del verbo, y el complemento indirecto (objeto indirecto) como la persona, animal o cosa en quien recae indirectamente la acción del verbo.

Oración [1] Frase verbal junto con las frases nominales o adverbiales u otras que dependan de ella. Las oraciones pueden ser dependientes o independientes. Por ejemplo, la oración independiente "Dice Juan que María te buscaba" contiene la oración dependiente "María te buscaba". A veces se marcan las oraciones poniéndolas entre corchetes. El estudio de la estructura de las oraciones es uno de los temas centrales de la sintaxis.

[2] A veces se usa en contraste con el término cláusula para indicar una oración independiente, con las cláusulas dependientes que pueda tener, o una serie de dos o más oraciones independientes coordenadas con una conjunción como "y" u "o". Compárese con enunciado.

Palabra Es una raíz, junto con los afijos que dependan de ella y posiblemente de otras raíces (en el caso de una raíz compuesta), que puede pronunciarse sola en el uso normal de una lengua, por ejemplo, como respuesta a una pregunta. Frecuentemente las palabras tienen rasgos fonológicos especiales. Compárese con frase, morfema.

Paradigma Lista de formas relacionadas de una palabra, especialmente si son relacionadas por flexión. Por ejemplo: "hablo, hablas, habla" es un paradigma de formas de tiempo presente singular del verbo "hablar"

	en el español; "hablar, hablante, hablado" es un paradigma de formas infinitiva del mismo verbo. La yuxtaposición de paradigmas paralelos en forma de un cuadro, puede facilitar la comparación y por lo tanto el análisis de las formas.
Parser	Véase *analizador sintáctico.*
Peso de similitud	El peso de similitud entre dos definiciones es un valor calculado por alguna medida de similitud o relación semántica. Este peso refleja cuan similares o parecidas pueden ser dos palabras, basándose en la definición de cada una.
Polisemia	Polisemia a la capacidad que tiene una sola palabra para expresar muy distintos significados. Pluralidad de significados de una palabra o de cualquier signo lingüístico y de un mensaje, con independencia de la naturaleza de los signos que lo constituyen.
Procesamiento de Lenguaje Natural (PLN)	Disciplina tiene por objetivo habilitar a las computadoras para que entiendan el texto, procesándolo por su sentido.
Raw data	Es un término utilizado para los datos sin procesar, conocidos como datos primarios.
Recuperación de información	La recuperación de información es la ciencia encargada de buscar información en archivos de diversos tipos, en meta-datos y en bases de datos textuales, de imágenes o de sonidos. La plataforma sobre la cual es posible realizar dichas búsquedas se extiende desde computadoras de escritorio, redes de computadoras privadas o públicas hasta intranets e Internet.
Semántica	La Semántica es el estudio de los significados de las estructuras de las lenguas (morfemas, palabras, frases, oraciones y otras). Una diferencia o semejanza semántica es una diferencia o semejanza de significados. Compárese con gramática, sintaxis, morfología,

fonología, léxico.

Sentido predominante — El sentido predominante de un vocablo es aquel que generalmente suele ser el más usado por los hablantes de una lengua específica.

SGML — SGML son las siglas de Standard Generalized Markup Language o "Lenguaje de Marcación Generalizado". Consiste en un sistema para la organización y etiquetado de documentos. La Organización Internacional de Estándares (ISO) ha normalizado este lenguaje en 1986. El lenguaje SGML sirve para especificar las reglas de etiquetado de documentos y no impone en sí ningún conjunto de etiquetas en especial.

Sintaxis — Es el estudio de cómo las palabras se combinan para formar frases y oraciones, ya sean dependientes o independientes. Compárese con gramática; contrástese con morfología, fonología, semántica, léxico.

Synset — Synset es un término (en inglés) usado en WordNet, donde hace referencia a un conjunto de sinónimos, comprendidos por palabras o colocaciones. Dicho conjunto se encuentra conectado con otros synsets mediante relaciones jerárquicas existentes en WordNet.

Terminales — En la nomenclatura BNF se le conoce como terminales a los nombres de las reglas sintácticas encerradas entre paréntesis.

Tesauro — El tesauro es un sistema que organiza el conocimiento basado en conceptos que muestran relaciones entre vocablos. Las relaciones expresadas comúnmente incluyen jerarquía, equivalencia y asociación (o relación). Los tesauros también proporcionan información como sinonimia, antonimia, homonimia, etc.

Token — Es un bloque de texto categorizado. Por ejemplo una marca de puntuación, un operador, un identificador, un número, etc.

Vecinos — Es un conjunto conformado por vocablos que mantienen cierta

	relación semántica con otro en específico. Dichos vocablos son seleccionados comparando diferentes contextos locales, de tal manera que los contextos más parecidos seleccionan a los vecinos.
Ventana de contexto	En desambiguación del sentido de las palabras, ventana de contexto se refiere a las palabras vecinas a la palabra ambigua. Dicha ventana es de tamaño variable, es decir, desde dos palabras, oración, párrafo, texto, etc.
WSD	De las siglas en inglés Word Sense Disambiguation en español desambiguación del sentido de las palabras. Es la tarea de seleccionar automáticamente un sentido, de un conjunto de posibilidades predefinidas, para una palabra dada en un texto.