



TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA INFORMÁTICA

# Diseño e implementación de un analizador de dependencias para procesamiento de lenguaje natural en Español

---

Mediante Máquinas de Soporte Vectoriales

**Autor**

Alejandro Alcalde Barros

**Directores**

Salvador García



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE  
TELECOMUNICACIÓN

—  
23 de noviembre de 2016

Alejandro Alcalde Barros: *Diseño e implementación de un analizador de dependencias para procesamiento de lenguaje natural en Español*, Mediante Máquinas de Soporte Vectoriales, Grado en Ingeniería Informática, © 23 de noviembre de 2016

DIRECTOR:  
Salvador García

LOCALIZACIÓN:  
Granada

ÚLTIMA MODIFICACIÓN:  
23 de noviembre de 2016

*Ohana* means family.  
Family means nobody gets left behind, or forgotten.  
— Lilo & Stitch

Dedicated to the loving memory of Rudolf Miede.  
1939–2005



## RESUMEN

---

En este trabajo se implementa un método para analizar dependencias palabra a palabra con una estrategia de abajo a arriba (*Bottom-Up*) usando Máquinas de Soporte Vectoriales (SVMs). En concreto, este trabajo se ha centrado en analizar las dependencias entre palabras en Castellano. Aunque la precisión de los resultados no está cerca del estado del arte, es necesario tener en cuenta que este parseador no usa información sobre la estructura de las frases.

ETIQUETAS: PNL, SVM, Parseo de dependencias.

## ABSTRACT

---

In this project, a method for analyzing word to word dependencies is implemented using a bottom-up strategy with the help of Support Vector machines. In particular, this work has focused in analyzing dependencies between words for spanish language. Even though accuracy is far from the state of the art, it is worth noting this parser is not using information about the sentences structure.

TAGS: NLP, SVM, Dependency parsing.



*We have seen that computer programming is an art,  
because it applies accumulated knowledge to the world,  
because it requires skill and ingenuity, and especially  
because it produces objects of beauty.*

---

## ACKNOWLEDGMENTS

---

Put your acknowledgments here.

Many thanks to everybody who already sent me a postcard!

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Jim Young, Lorenzo Pantieri and Enrico Gregorio<sup>1</sup>, Jörg Sommer, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, Roland Winkler, Jörg Weber, Henri Menke, Claus Lahiri, Clemens Niederberger, Stefano Bragaglia, Jörn Hees, and the whole L<sup>A</sup>T<sub>E</sub>X-community for support, ideas and some great software.

*Regarding L<sub>Y</sub>X:* The L<sub>Y</sub>X port was initially done by *Nicholas Mariette* in March 2009 and continued by *Ivo Pletikosić* in 2011. Thank you very much for your work and for the contributions to the original style.

---

<sup>1</sup> Members of GuIT (Gruppo Italiano Utilizzatori di T<sub>E</sub>X e L<sup>A</sup>T<sub>E</sub>X)





# ÍNDICE GENERAL

---

<b>I</b>	<b>PUESTA EN ESCENA</b>	<b>1</b>
<b>1</b>	<b>MOTIVACIÓN E INTRODUCCIÓN</b>	<b>3</b>
1.1	¿Qué es el Procesamiento del Lenguaje Natural?	3
1.2	Historia del Procesamiento del Lenguaje Natural	7
1.3	Limitaciones	8
1.4	El pipeline genérico	9
1.4.1	Pasos previos	10
1.4.2	Proceso principal del análisis de sentimientos	13
1.5	El pipeline de CORENLP	15
1.6	Software Existente	18
1.7	Estado del arte	18
1.7.1	A nivel del documento	18
1.7.2	A nivel de sentencia	18
1.7.3	A nivel de entidad	19
<b>II</b>	<b>OBJETIVOS</b>	<b>21</b>
<b>2</b>	<b>PARSEO DE DEPENDENCIAS PARA ESPAÑOL</b>	<b>23</b>
<b>III</b>	<b>RESOLUCIÓN DEL TRABAJO</b>	<b>25</b>
<b>IV</b>	<b>CONCLUSIONES Y VÍAS FUTURAS</b>	<b>27</b>
<b>V</b>	<b>APPENDIX</b>	<b>29</b>
	<b>BIBLIOGRAFÍA</b>	<b>31</b>








## ÍNDICE DE FIGURAS

---

Figura 1	Ejemplo de parseo de dependencias	8
Figura 2	Ejemplo de parseo de dependencias 2	9
Figura 3	Visualización del <i>pipeline</i> de CORENLP	17

## TODO LIST

---

	Describir de forma más detallada el Dep Parsing, mencionar state of the art (3/4 papers), entre ellos el implementado .	3
	Sección con algoritmo implementado, reiterando sección anterior pero con lujo de detalles (Teóricos y código) . . . . .	3
	Añadir sección “El resto del paper está organizado...” . . . . .	3
	¿Pongo ejemplos? <i>To give a few examples, in [47], Pang and Lee attempted to predict the polarity of movie reviews using three different machine learning techniques: Naïve Bayes, Maximum Entropy classification and Support Vector Machine (SVM). Similarly, in [50] the same authors tried to predict the rating of a movie given in a review, instead of just classifying the review into a positive or negative class.</i> . . . . .	14
	Pongo ejemplos? . . . . .	15
	De nuevo, pongo ejemplos? . . . . .	15
	Menciono software existente? . . . . .	18

## ÍNDICE DE CUADROS

---

Cuadro 1	<i>Pipeline</i> de CORENLP y disponibilidad por lenguaje	10
----------	--	----

## LISTINGS

---

## ACRONYMS

---

NLP	Natural Language Processing
PNL	Procesamiento del Lenguaje Natural
IA	Inteligencia Artificial
ASR	Automatic Speech Recognition
RVA	Reconocimiento de Voz Automático
POS	Part-Of-Speech
AA	Aprendizaje Automático
NER	Named Entities
DP	Dependency Parsing
HRL	High-Resource Language
LRL	Low-Resource Language
SDS	Spoken Dialogue System
DM	Dialogue Management
TTS	Text-To-Speech
API	Application Programming Interface
HTML	Hyper Text Markup Language
CRF	Conditional Random fields
PTB	Penn Tree Bank
XML	Extensible Markup Language
SO	Semantic Orientation
PMI-IR	Pointwise Mutual Information and Information Retrieval
SVM	Support Vector Machine



## Parte I

### PUESTA EN ESCENA

En este primer apartado expondremos el conocimiento previo necesario para que el lector se situe en el contexto del problema tratado.



## MOTIVACIÓN E INTRODUCCIÓN

Describir de forma más detallada el Dep Parsing, mencionar state of the art (3/4 papers), entre ellos el implementado

Sección con algoritmo implementado, reiterando sección anterior pero con lujo de detalles (Teóricos y código)

Añadir sección "El resto del paper está organizado..."

### 1.1 ¿QUÉ ES EL PROCESAMIENTO DEL LENGUAJE NATURAL?

El lenguaje natural se refiere a cualquier lenguaje hablado por un humano [19], (por ejemplo, Inglés, Castellano o Chino). El *Natural Language Processing* (NLP) es un campo de la ciencia de la computación e ingeniería desarrollado a partir del estudio del lenguaje y la computación lingüística dentro del campo de la Inteligencia Artificial (IA). Los objetivos del NLP son diseñar y construir aplicaciones que faciliten la interacción humana con la máquinas y otros dispositivos mediante el uso del lenguaje natural. Dentro del amplio campo del NLP podemos distinguir las siguientes áreas principales:

Procesamiento del  
Lenguaje  
Natural (PNL)

El lenguaje natural se refiere a cualquier lenguaje hablado por un humano, (por ejemplo, Inglés, Castellano o Chino). El NLP es un campo de la ciencia de la computación e ingeniería desarrollado a partir del estudio del lenguaje y la computación lingüística dentro del campo de la IA. Los objetivos del NLP son diseñar y construir aplicaciones que faciliten la interacción humana con la máquinas y otros dispositivos mediante el uso del lenguaje natural. Dentro del amplio campo del NLP podemos distinguir las siguientes áreas principales [7, 19]:

Procesamiento del  
Lenguaje  
Natural (PNL)

**RESÚMENES** este área incluye aplicaciones que puedan, basándose en una colección de documentos, dar como salida un resumen coherente del contenido de los mismos. Otra de las posibles aplicaciones sería generar presentaciones a partir de dichos documentos. En los últimos años, la información disponible en la red ha aumentado considerablemente. Un claro ejemplo es la literatura científica, o incluso repositorios de información más genérica como *Wikipedia*. Toda esta información escrita en lenguaje natural puede aprovecharse para entrenar modelos que sean capaces de generar hipótesis por sí mismos, generar resúmenes o extraer hechos. Un ejemplo claro puede ser la extracción de hechos básicos que relacionen dos entidades ("Luís es padre de Cristina").

**TRADUCCIÓN AUTOMÁTICA:** Esta fue la principal área de investigación en el campo del [NLP](#). Como claro ejemplo tenemos el traductor de Google, mejorando día a día. Sin embargo, un traductor realmente útil sería aquel que consiga traducir en tiempo real una frase que le dictemos mientras decidimos qué línea de autobús debemos coger para llegar a tiempo a una conferencia en Zurich. La traducción entre lenguajes es quizá una de las formas más transcendentales en las que las máquinas podrían ayudar en comunicaciones entre humanos. Además, la capacidad de las máquinas para traducir entre idiomas humanos se considera aún como un gran test a la [IA](#), ya que una traducción correcta no consiste en el mero hecho de generar frases en un idioma humano, también requiere del conocimiento humano y del contexto, pese a las ambigüedades de cada idioma. Por ejemplo, la traducción literal “*bordel*” en Francés significa Burdel; pero si alguien dice “*Mi cuarto es un bordel*”, el traductor debería tener el conocimiento suficiente para inferir que la persona se está refiriendo a que su habitación es un desorden.

Conocido como texto  
paralelo

La traducción automática fue una de las primeras aplicaciones no numéricas de la computación y comenzó a estudiarse de forma intensiva en la década de los 50. Sin embargo, no fue hasta la década de los 90 cuando se produjo una transformación en este área. IBM se hizo con una gran cantidad de frases en Inglés y Francés que eran traducciones las unas de las otras, lo cual permitió recopilar estadísticas de traducciones de palabras y secuencias de palabras, concediendo así el desarrollo de modelos probabilísticos para la traducción automática. Hasta ese momento, todo el análisis gramático se hacía manualmente.

A la llegada del nuevo milenio, se produjo una explosión de texto disponible en la red, así como grandes cantidades de *texto paralelo*. Se dieron invención a nuevos sistemas para la traducción automática basados en modelos estadísticos basados en frases en lugar de palabras. En lugar de traducir palabra por palabra, ahora se tenían en cuenta pequeños grupos de palabras que a menudo poseen una traducción característica.

En los últimos años, y mediante el uso de *deep learning* se están desarrollando modelos de secuencias basados en este tipo de aprendizaje bastante prometedores. La idea principal del *deep learning* reside en entrenar un modelo con varios niveles de representación para optimizar el objetivo deseado, una traducción de calidad, en este caso. Mediante estos niveles el modelo puede aprender representaciones intermedias útiles para la tarea que le ocupa. Este método de aprendizaje se ha explotado sobre todo en redes neuronales. Un ejemplo claro de *deep learning* usando redes neuronales es el reconocimiento de dígitos, cada capa interna de la red neuronal intenta extraer características representativas de cada dígito a distintas escalas. Podemos ver una demostración de este comportamiento en Pound y Riley [18]



**RECONOCIMIENTO DE VOZ:** Una de las tareas más difíciles en [NLP](#). Aún así, se han conseguido grandes avances en la construcción de modelos que pueden usarse en el teléfono móvil o en el ordenador. Estos modelos son capaces de reconocer expresiones del lenguaje hablado como preguntas y comandos. Desafortunadamente, los sistemas *Automatic Speech Recognition* ([ASR](#)) funcionan bajo dominios muy acotados y no permiten al interlocutor desviarse de la entrada que espera el sistema, por ejemplo, “*Por favor, diga ahora la opción a elegir: 1 Para... , 2 para...*”

*Reconocimiento de  
Voz  
Automático ([RVA](#))*

**SDS:** Los *Spoken Dialogue Systems* ([SDSs](#)). El diálogo ha sido un tema popular para el [NLP](#) desde los 80. En estos sistemas se pretende reemplazar a los usuales buscadores en los que introducimos un texto para obtener algún tipo de respuesta a una pregunta. Por ejemplo, si quisieramos saber a qué hora abre un centro comercial, bastaría con hablarle al sistema en lenguaje natural – nuestro lenguaje natural, ya sea Inglés, Alemán o Castellano y el sistema nos daría respuesta a nuestra pregunta. Aunque ya existen este tipo de sistemas (por ejemplo, *Siri de Apple*, *Cortana de Microsoft*, *Google Now...*) están aún en una situación muy precaria, ya que ninguno entiende por completo el lenguaje natural, solo un subconjunto de frases clave.

*[SDS](#): Sistemas de  
Diálogo Hablados*

La creación de [SDSs](#), ya sea entre humanos o entre humanos y agentes artificiales requiere de herramientas como:

- [ASR](#), para identificar qué dice el humano.
- *Dialogue Management* ([DM](#)), para determinar qué quiere el humano.
- Acciones para obtener la información o realizar la actividad solicitada.
- Síntesis *Text-To-Speech* ([TTS](#)), para comunicar dicha información al humano de forma hablada.

*[DM](#): Gestión del  
diálogo*

*Leer un texto por  
una máquina*

En la actualidad, Hinton y col. [6] desarrollaron un [SDS](#) haciendo uso de *deep learning* para asociar señales sonoras a secuencias de palabras y sonidos del idioma humano, logrando avances importantes en la precisión del reconocimiento del habla.

**CLASIFICACIÓN DE DOCUMENTOS:** Una de las áreas más exitosas del [NLP](#), cuyo objetivo es identificar a qué categoría debería pertenecer un documento. Ha demostrado tener un amplio abanico de aplicaciones, por ejemplo, filtrado de *spam*, clasificación de artículos de noticias, valoraciones de películas... Parte de su éxito e impacto se debe a la facilidad relativa que conlleva entrenar los modelos de aprendizaje para hacer dichas clasificaciones.

**ANÁLISIS DE SENTIMIENTOS:** Gran parte del trabajo en [NLP](#) se ha centrado en el análisis de sentimientos (identificación de orientaciones positivas o negativas en textos) e identificación de creencias positivas, negativas o neutrales en frases basándose en información léxica y sintáctica. Tanto las creencias como los sentimientos constituyen actitudes hacia eventos y proposiciones, aunque en concreto, los sentimientos pueden también referirse a actitudes hacia objetos tales como personas, organizaciones y conceptos abstractos. La detección de sentimientos y emociones en texto requiere de información léxica y a nivel de la propia sentencia. Por lo general, el sentimiento puede detectarse a través del uso de palabras expresando orientaciones positivas o negativas, por ejemplo, *triste*, *preocupado*, *difícil* son todas palabras con una connotación negativa, mientras que *cómodo*, *importante*, *interesante* denotan un sentimiento positivo. Las aproximaciones más sofisticadas para el análisis de sentimientos intentan buscar tanto la fuente como el objeto del sentimiento, por ejemplo, quién está expresando un sentimiento positivo sobre alguna persona, objeto, actividad o concepto.

La comunidad del reconocimiento de voz está igualmente implicada en el estudio de actitudes positivas y negativas, centrándose en la identificación de emociones haciendo uso de información acústica y prosódica, es decir un relieve en la pronunciación. Otras investigaciones se han centrado en identificar emociones particulares, específicamente las seis emociones básicas según Ekman – ira, aversión, temor, dicha, tristeza y asombro – las cuales pueden ser reacciones a eventos, proposiciones u objetos. Por contra, la generación de emociones ha demostrado ser un reto mucho mayor para la síntesis [TTS](#).

La clasificación de sentimientos es algo ampliamente usado para identificar opiniones – puntos de vista positivos o negativos hacia personas, instituciones o ideas – en muchos idiomas y géneros. Una de las aplicaciones más prácticas, y de las que más abundan consiste en identificar críticas sobre películas o productos [[16](#), [27](#)].

La minería de datos en redes sociales con el fin de realizar análisis de sentimientos se ha convertido en un tema popular con el objetivo de evaluar el *estado de ánimo* del público – de twitter, por ejemplo. –

El [NLP](#) emplea técnicas computacionales con el propósito de aprender, entender y producir lenguaje humano. Las aproximaciones de hace unos años en el campo de la investigación del lenguaje se centraban en automatizar el análisis de las estructuras lingüísticas y desarrollar tecnologías como las mencionadas anteriormente. Los investigadores de hoy en día se centran en usar dichas herramientas en aplicaciones para el mundo real, creando sistemas de diálogo hablados y motores de traducción *Speech-to-Speech*, es decir, dados dos interlocutores, interpretar y traducir sus frases. Otro de los focos en los que se centran las investigaciones actuales son la minería en redes sociales en busca

de información sobre salud, finanzas e identificar los sentimientos y emociones sobre determinados productos.

## 1.2 HISTORIA DEL PROCESAMIENTO DEL LENGUAJE NATURAL

A continuación, citamos algunos de los avances en este campo durante los últimos años según Hirschberg y Manning [7].

Durante las primeras épocas de esta ciencia, se intentaron escribir vocabularios y reglas del lenguaje humano para que el ordenador las entendiera. Sin embargo, debido a la naturaleza ambigua, variable e interpretación dependiente del contexto de nuestro lenguaje resultó una ardua tarea. Por ejemplo, una estrella puede ser un ente astronómico o una persona, y puede ser un nombre o un verbo.

En la década de los 90, los investigadores transformaron el mundo del NLP desarrollando modelos sobre grandes cantidades de datos sobre lenguajes. Estas bases de datos se conocen como *corpus*. El uso de estos conjuntos de datos fueron uno de los primeros éxitos notables del uso del *big data*, mucho antes de que el Aprendizaje Automático (AA) acuñara este término.

Esta aproximación estadística al NLP descubrió que el uso de métodos simples usando palabras, secuencias del *Part-Of-Speech* (POS) (si una palabra es un nombre, verbo o preposición), o plantillas simples pueden obtener buenos resultados cuando son entrenados sobre un gran conjunto de datos. A día de hoy, muchos sistemas de clasificación de texto y sentimientos se basan únicamente en los distintos conjuntos de palabras o “*bag of words*” que contienen los documentos, sin prestar atención a su estructura o significado. El estado del arte de hoy día usa aproximaciones con AA y un rico conocimiento de la estructura lingüística. Un ejemplo de estos sistemas es *Stanford CORENLP* [14]. CORENLP proporciona un *pipeline* estándar para el procesamiento del NLP incluyendo:

POS: Categorías morfosintácticas en castellano

**POS TAGGING:** Etiquetado morfosintáctico. Módulo encargado de leer texto en algún lenguaje y asignar la categoría morfosintáctica a cada palabra, por ejemplo, nombre, verbo, adjetivo... aunque por lo general se suelen usar etiquetas más detalladas como “*nombre-plural*”.

**NER:** *Named Entities* (NER), etiqueta palabras en un texto correspondientes a *nombres de cosas*, como personas, nombres de compañías, nombres de proteínas o genes etc. En concreto, CORENLP distingue de forma muy precisa tres tipos de clases, personas, organizaciones y localizaciones.

**PARSEO GRAMATICAL:** Resuelve la estructura gramatical de frases, por ejemplo, qué grupos de palabras van juntos formando frases y qué palabras son sujeto u objeto de un verbo. Como se ha comentado,

en aproximaciones anteriores se usaban parseadores probabilísticos usando conocimiento del lenguaje a partir de sentencias analizadas sintácticamente a mano. Para así producir el análisis más probable de sentencias nuevas. Actualmente se usan parseadores estadísticos, los cuales aún comenten fallos, pero funcionan bien a rasgos generales.

**DP:** *Dependency Parsing* (**DP**) o parseo de dependencias. Analiza la estructura gramatical de una frase, estableciendo relaciones entre palabras principales y palabras que modifican dichas palabras principales. La [Figura 1](#) muestra un ejemplo. La flecha dirigida de la palabra *moving* a la palabra *faster* indica que *faster* modifica a *moving*. La flecha está etiquetada con una palabra, en este caso *advmod*, indicando la naturaleza de esta dependencia. La [Figura 2](#) muestra ejemplos de los distintos módulos del *pipeline* de CORENLP

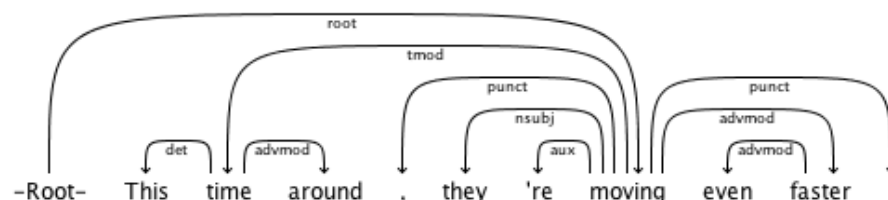


Figura 1: Ejemplo de parseo de dependencias

### 1.3 LIMITACIONES

Aunque se han producido avances, una de las principales limitaciones del **NLP** hoy día es el hecho de que la mayoría de recursos y sistemas solo están disponibles para los denominados *High-Resource Languages* (**HRLs**), estos lenguajes son el Inglés, Francés, Español, Alemán y Chino. Por contra, hay una gran cantidad de *Low-Resource Languages* (**LRLs**) – como Bengalí, Indonesio, Punjabí, Cebuano y Swahili – hablados y escritos por millones de personas que no disponen de este tipo de sistemas. Uno de los mayores retos para la comunidad del lenguaje es desarrollar recursos y herramientas para cientos o miles de lenguajes, no solo para unos pocos.

**HRL:** Idiomas de  
altos recursos

**LRL:** Idiomas de  
bajos recursos

Aún existiendo bastante *software* trabajando con **NLP**, y para idiomas **HRL**, suelen obtenerse mejores resultados para un idioma en concreto, el Inglés. Es por ello que este trabajo se ha centrado en desarrollar una fase del *pipeline* que se encuentra en todos los sistemas que realizan análisis de sentimientos, y en general **NLP** para el idioma Español. Como ejemplo podemos citar el famoso CORENLP [14].

En la [Tabla 1](#) se lista todo el *pipeline* de CORENLP junto con el soporte para cada lenguaje. Como se aprecia, el *pipeline* está completo únicamente para el Inglés. El objetivo de este trabajo ha consistido en implementar un parseo de dependencias para el Español.

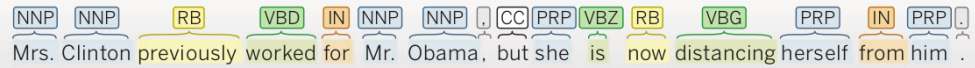
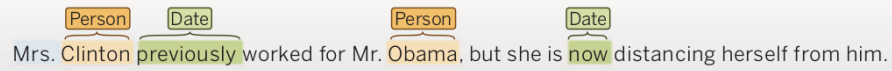
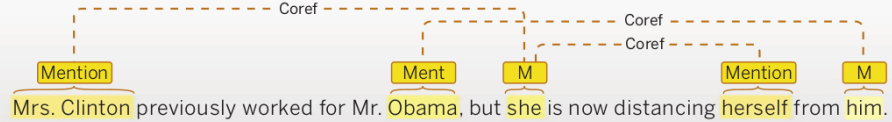
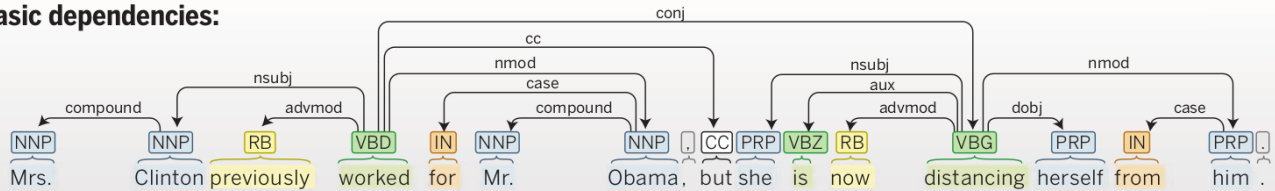
**Part of speech:****Named entity recognition:****Co-reference:****Basic dependencies:**

Figura 2: Ejemplo de la salida del programa CORENLP. De arriba a abajo, se muestra la categoría morfosintáctica de cada palabra, el nombre de algunas entidades, determina qué entidad hace co-referencia a la misma persona u organización y por último la estructura sintáctica de cada frase, usando un análisis de dependencias gramaticales.

Con la introducción del *pipeline* de CORENLP, se pasa ahora a describir el proceso que todo sistema para NLP debe seguir. Comenzaremos mencionando un proceso genérico, para después profundizar en el *pipeline* de un *software* específico, CORENLP en nuestro caso.

## 1.4 EL PIPELINE GENÉRICO

En esta sección se comentará el proceso habitual que suele seguirse como *pipeline* en los problemas de NLP. Para ello se describirán los distintos niveles de análisis en los que opera dicho *pipeline*, así como las diferentes aproximaciones que se usan y los problemas más comunes a los que se enfrenta todo sistema que realice NLP.

Liu [12] define una opinión como una quintupla conteniendo el objetivo de la opinión (o *entidad*), el atributo del objetivo al que se dirige la opinión, el sentimiento (o polaridad) de la opinión, pudiendo ser este positivo, negativo o neutral, el poseedor de dicha opinión y la fecha en la que se produjo. Formalmente se podría definir como la tupla:

$$(e_i, a_{ij}, s_{ijkl}, h_k, t_l)$$

donde  $e_i$  corresponde con el objetivo de la opinión  $i$ -ésima,  $a_{ij}$  es el  $j$ -ésimo atributo de  $e_i$ ,  $h_k$  el  $k$ -ésimo poseedor de la opinión,  $t_l$

Cuadro 1: *Pipeline* de CORENLP y disponibilidad por lenguaje

ANNOTATOR	AR	ZH	EN	FR	DE	ES
Tokenize / Segment	✓	✓	✓	✓		✓
Sentence Split	✓	✓	✓	✓	✓	✓
Part of Speech	✓	✓	✓	✓	✓	✓
Lemma			✓			
Named Entities		✓	✓		✓	✓
Constituency Parsing	✓	✓	✓	✓	✓	✓
Dependency Parsing		✓	✓	✓	✓	
Sentiment Analysis			✓			
Mention Detection		✓	✓			
Coreference		✓	✓			
Open IE			✓			

codifica el tiempo en el que se emitió la opinión y por último,  $s_{ijkl}$  es la polaridad de la opinión hacia el atributo  $a_{ij}$  para la entidad  $e_i$  por el poseedor de la opinión  $h_k$  en el momento  $t_l$ .

El principal objetivo del análisis de sentimientos consiste en encontrar todas las tuplas  $(e_i, a_{ij}, s_{ijkl}, h_k, t_l)$  en un documento o colección de documentos.

#### 1.4.1 Pasos previos

El procesamiento más usual para realizar tareas de análisis de sentimientos se puede dividir en una serie de pasos definidos. Dichos pasos corresponden a la adquisición del *corpus* o datos, preprocesamiento del texto, el proceso principal del análisis de sentimientos, agregación y resumen de los resultados y por último, visualización. En los próximos apartados se mencionarán los tres primeros.

##### 1.4.1.1 Adquisición de Datos

En este paso se debe obtener el *corpus* para el cual se desea realizar el análisis de sentimientos. Actualmente existen dos aproximaciones para realizar esta tarea. Una de ellas consiste en hacer uso de la *Application Programming Interface* (API) de alguna página web de la que se desee extraer el *corpus*. Una de las APIs más populares para este propósito es la de Twitter. La segunda aproximación hace uso de *Web Crawlers* para extraer datos de las webs deseadas.

Rastreadores de  
Webs

Ambas aproximaciones presentan sus ventajas y desventajas, y por tanto se debe encontrar un equilibrio en función de cual se decida usar. Veamos algunos de ellos.

Mediante el uso de una [API](#) la implementación es sencilla, los datos obtenidos están ordenados y poseen una estructura poco sujeta al cambio, sin embargo, en función del proveedor de la [API](#) se presentan ciertas limitaciones. Siguiendo con Twitter, su [API](#) limita a 180 consultas cada 15 minutos el número de peticiones que se pueden realizar. Además, su [API](#) para *streaming* presenta otras limitaciones. En lugar de imponer límites a la cantidad de peticiones, restringe el número de clientes que se pueden conectar desde la misma dirección IP al mismo tiempo, así como la velocidad a la que cada uno puede leer los datos. Pese a las limitaciones anteriores, la más importante quizás sea que esta aproximación depende de la existencia de una [API](#) por parte del sitio web.

Por otro lado, la aproximación basada en rastreadores webs son bastante más complejas de implementar, la razón principal se debe a que los datos obtenidos, por norma general tendrán ruido y no estarán estructurados. Como beneficio, esta aproximación tiene la capacidad no imponernos prácticamente ninguna restricción. Si bien es cierto que se deben respetar ciertas normas y protocolos, como las indicaciones del fichero `ROBOTS.TXT`<sup>1</sup> de cada sitio web, no realizar múltiples peticiones al mismo servidor y espaciar las mismas para no someter al servidor a demasiada carga.

#### 1.4.1.2 Preprocesamiento del texto

El segundo paso en el *pipeline* del análisis de sentimientos es el preprocesamiento del texto adquirido. En este paso se realizan varias tareas habituales para el [NLP](#) correspondientes al análisis léxico. Algunas de estas tareas son:

**TOKENIZACIÓN:** Es una técnica fundamental para la mayoría de tareas en [NLP](#). Encargada de separar las cadenas de texto del documento completo en una lista de palabras. Es muy sencilla de realizar para idiomas delimitados por espacios como el Inglés, Español o Francés, pero se torna considerablemente más compleja para idiomas donde las palabras no son delimitadas por espacios, como el Japonés, Chino y Thai.

Los idiomas anteriores requieren de un proceso llamado segmentación de palabras, el cual es un problema de etiquetado secuencial. Para resolverlo se usan *Conditional Random fields* ([CRF](#)), método que ha demostrado ser superior a los modelos de Markov ocultos y modelos de Markov de máxima entropía [10, 17, 23]. Debido a que es una técnica fundamental, existen multitud de herramientas disponibles, para idiomas delimitados por espacios, el tokenizador de Stanford<sup>2</sup>

<sup>1</sup> <http://www.robotstxt.org/robotstxt.html>

<sup>2</sup> <http://nlp.stanford.edu/software/tokenizer.shtml>



o el de OPENNLP<sup>3</sup>. Para la segmentación de palabras en Chino, existen herramientas como ICTCLAS<sup>4</sup>, THULAC<sup>5</sup> y el segmentador de Stanford<sup>6</sup>.

**STEMMING:** Proceso heurístico encargado de eliminar los afijos de la palabra para dejarlos en su forma canónica (invariante, o raíz). Por ejemplo, *persona*, *personificar* y *personificación* pasan a ser *persona* una vez acabado este proceso.

**LEMATIZACIÓN:** Proceso algorítmico para convertir una palabra a su forma de diccionario no-inflexible – *non-inflected* en inglés –. Esta fase es análoga a la anterior (*stemming*) pero se realiza a través de una serie de pasos más rigurosos que incorporan un análisis morfológico de cada palabra.

**ELIMINACIÓN DE STOPWORDS:** Actividad encargada de borrar las palabras usadas para estructurar el lenguaje pero que no contribuyen de modo alguno a su contenido. Algunos ejemplos de estas palabras pueden ser *de*, *la*, *que*, *el*, *en*, *y*, *a*, *los*, *del*, *se*, *las*, *por*, *un*, *par*, *con*.<sup>7</sup>

**SEGMENTACIÓN DE FRASES:** Procedimiento que separa párrafos en sentencias. Presenta sus propios retos, ya que los signos de puntuación, como el punto (.) se usan con frecuencia para marcar tanto el fin de una frase como para denotar abreviaciones y números decimales.

**POS TAGGING Y PARSEO** o etiquetado morfosintáctico. Paso que etiqueta cada palabra de una sentencia con su categoría morfosintáctica, como *adjetivo*, *nombre*, *verbo*, *adverbio* y *preposición*. Estas etiquetas pueden usarse como entrada para procesamientos futuros, como el parseo de dependencias (Objetivo de este trabajo) o como característica para el proceso de AA. De igual manera que la segmentación, es un problema de etiquetado secuencial. El etiquetado morfosintáctico proporciona información léxica, el parseo obtiene información sintáctica. El parseo genera un árbol que representa la estructura gramatical de una sentencia dada con la correspondiente relación entre los distintos constituyentes. Este trabajo se ha centrado en construir un parseo de dependencias para el Español.

3 <https://opennlp.apache.org/documentation/manual/opennlp.html#tools.tokenizer>

4 <http://ictclas.nlpir.org>

5 <http://thulac.thunlp.org>

6 <http://nlp.stanford.edu/software/segmenter.shtml>

7 Para ver una lista completa de palabras visitar: <http://snowball.tartarus.org/algorithms/spanish/stop.txt>



Cabe destacar que no es obligatorio aplicar todos y cada uno de los pasos anteriores. En función del tipo de aplicación se ejecutarán unos pasos u otros. Por ejemplo, un sistema basado en [AA](#) probablemente aplicará cada uno de estos pasos con el fin de reducir la dimensionalidad y ruido del problema. Por contra, una aproximación no-supervisada quizá necesite la categoría sintáctica de algunas de las *stopwords* para construir reglas de dependencia con el fin de usarlas posteriormente en el proceso principal del análisis. Es claro pues, que la aproximación no-supervisada en este caso deberá omitir la fase eliminación de *stopwords*. En [Otras aproximaciones](#) se describe con más detalle las diferencias entre aproximaciones supervisadas frente a no supervisadas.

Por otra parte, existen otro tipo de pasos dependientes en su totalidad del origen de los datos y el método de adquisición. En particular, los datos que se obtengan a través de un rastreador web deberán ser procesados con fin de eliminar las etiquetas *Hyper Text Markup Language* ([HTML](#)) e información no textual – como imágenes y anuncios – El texto extraído de Twitter necesitará de especial atención en cuanto a *hashtags*, menciones, *retweets*, texto póbaramente escrito, emoticonos, carcajadas escritas y palabras con caracteres repetidos — *siiiiiiiiiii* —

#### 1.4.2 Proceso principal del análisis de sentimientos

La tercera fase en el *pipeline* es el proceso principal del análisis. A continuación se mencionarán los distintos niveles de granularidad en los que actúan las aproximaciones más comunes.

##### 1.4.2.1 Niveles de análisis

Desde que el análisis de sentimientos comenzó a ganar popularidad, se han ido proponiendo distintos niveles para el análisis en las distintas etapas. La primera se realizaba a nivel del documento, donde el objetivo residía en identificar la polaridad general del mismo. Más tarde, el interés se desplazó hacia un nivel más específico, las setencias. Por último, se bajó un paso más en cuanto a la granularidad para interesarse a nivel de entidad. Cabe destacar que los niveles más granulares pueden agruparse o conglomerarse para formar niveles más altos – menos granulares, a mayor escala – Por ejemplo, una análisis de sentimientos podría calcular la media de polaridades en una frase y producir un resultado a nivel de sentencias. Veamos a continuación los distintos niveles.

**NIVEL DE DOCUMENTO:** A este nivel, el análisis trata de clasificar el documento al completo con una polaridad positiva o negativa. La utilidad de este nivel a menudo está limitada y por normal general se usa en el contexto del análisis de reseñas [\[13\]](#). Formalmente, el objetivo para este tipo de tareas puede definirse como una versión

modificada de la representación introducida en la [Sección 1.4](#) y corresponde a la búsqueda de tuplas

$$(-, \text{GENERAL}, S_{\text{GENERAL}}, -, -)$$

donde la entidad  $e$ , el poseedor de la opinión  $h$ , y el tiempo  $t$  en el que se manifestó la opinión se asumen conocidos o se ignoran. El atributo  $a_j$  de la entidad  $e$  se corresponde con GENERAL. Todo esto implica que el análisis devolverá solamente la polaridad general del documento.

¿Pongo ejemplos?

*To give a few examples, in [47], Pang and Lee attempted to predict the polarity of movie reviews using three different machine learning techniques: Naïve Bayes, Maximum Entropy classification and Support Vector Machine (SVM). Similarly, in [50] the same authors tried to predict the rating of a movie given in a review, instead of just classifying the review into a positive or negative class.*

**NIVEL DE SENTENCIA:** Análogo al anterior, ya que se podría considerar una sentencia como un documento corto. Sin embargo, este nivel presenta algunos pasos de preprocesamiento consistentes en separar el documento en oraciones, paso que a su vez posee retos similares a la tokenización de idiomas no delimitados por periodos — visto en [Preprocesamiento del texto](#) —

**NIVEL DE ENTIDAD Y ASPECTO:** El más granular de todos a los que el análisis de sentimientos puede trabajar. A este nivel la tarea no consiste únicamente en encontrar la polaridad de una opinión, también su objetivo — a quién va dirigida — Por esto mismo, la definición de la quintupla en la [Sección 1.4](#) aplica al completo. El análisis a nivel de documento y sentencias funcionan bien cuando el texto analizado contiene una sola entidad y aspecto, pero empeoran para varios [4]. Para resolver este tipo de problemas, algunos sistemas de análisis de sentimientos basados en aspectos intentan detectar cada aspecto mencionado en el texto para asociarlo con una opinión.

El primer trabajo que se ocupó de resolver este problema fue obra de Hu y Liu [8]. Hu y Liu detectaban características de productos – aspectos – comentados con frecuencia por clientes, luego identificaban dichas sentencias con opiniones, las evaluaban en base a su polaridad y finalmente resumían los resultados.

#### 1.4.2.2 Otras aproximaciones

Existen dos aproximaciones para llevar a cabo el proceso de análisis de sentimientos. Una basada en léxico, no supervisada. Esta aproximación depende de reglas y heurísticas obtenidas del conocimiento lingüístico [26]. La otra aproximación es supervisada, usa [AA](#), aquí se utilizan algoritmos que aprenden la información subyacente de datos previamente anotados, permitiéndoles así clasificar instancias nuevas — sin etiquetar [16] —. Aunque estas dos aproximaciones son las más usadas, existen estudios que han demostrado buenos resultados al combinar ambas. Pasamos ahora a describirlas en detalle.

**APROXIMACIÓN NO SUPERVISADA BASADA EN EL LÉXICO:** También llamada basada en la semántica. Intenta determinar la polaridad

del texto usando un conjunto de reglas y heurísticas obtenidas del conocimiento del idioma. Los pasos habituales consisten en marcar primero cada palabra y frase con su correspondiente polaridad con ayuda de un diccionario. El siguiente paso incorpora el análisis de modificadores de sentimientos – como la negación – y su ámbito – intensificadores y negación –. Por último se tratan las conjunciones adversativas – *pero, aunque, mas* – comprendiendo cómo afectan a la polaridad y reflejándolo en la puntuación final del sentimiento [13].

Pongo ejemplos?

**APROXIMACIÓN SUPERVISADA BASADA EN APRENDIZAJE:** Igualmente conocida como aproximación basada en **AA** o métodos estadísticos para la clasificación de sentimientos. Consiste en algoritmos que aprenden los patrones subyacentes de los datos de entrenamiento — datos cuya clase o etiqueta se conoce para cada instancia — para después intentar clasificar nuevos datos suministrados al algoritmo, esta vez sin estar etiquetados. Los pasos a seguir en una aproximación de este tipo consisten en realizar algo de ingeniería de características que representen el objeto cuya clase se quiere predecir. Tras esto, se usan dichas representaciones como entrada del algoritmo. Algunas de las características más usadas en el análisis de sentimientos son: frecuencia del término, categorías morfosintáctica – **POS tags** – palabras y frases con sentimientos, reglas de opinión, modificadores del sentimiento y dependencias sintácticas, por mencionar algunas [13].

De nuevo, pongo ejemplos?

**APROXIMACIÓN BASADA EN CONCEPTOS:** Relativamente moderna. Consiste en usar *ontologías* para apoyar la tarea del análisis de sentimientos. Las ontologías suelen presentarse como gráfos donde los conceptos se asocian a nodos enlazados por relaciones. El estudio realizado por Zhou y Chaovalit [30] analiza en profundidad las ontologías, así como sus aplicaciones y desarrollo.

Una de las ventajas de usar métodos no supervisados reside en la no dependencia de grandes cantidades de datos para entrenar a los algoritmos. Aún así, sigue siendo necesaria la construcción u obtención de un léxico para los sentimientos. Los métodos no supervisados son menos dependientes del dominio que los métodos supervisados. De hecho, los clasificadores entrenados en un dominio específico muestran de forma consistente peor comportamiento cuando son ejecutados en otros dominios [1].

*Una ontología se define como un modelo que conceptualiza el conocimiento de un dominio dado, de tal forma que pueda ser comprendido tanto por humanos como máquinas.*

## 1.5 EL PIPELINE DE CORENLP

En la [Sección 1.4](#) se ha visto el proceso genérico a seguir para problemas de **NLP**, se presenta ahora una breve descripción de los pasos

que se mostraron en la [Tabla 1](#), correspondientes al *pipeline* para un software concreto — CORENLP [14] — Como se aprecia, dicho *pipeline* solo está totalmente completo para el Inglés.

CORENLP viene empaquetado con los modelos para Inglés. Es posible descargar modelos para distintos idiomas, pero por separado. El soporte para estos idiomas no es completo. Los pasos del *pipeline* mencionados a continuación se centran en la versión para el Inglés. En [Otras aproximaciones](#) se citaron los distintos modos de realizar tareas para NLP, los modelos de CORENLP entrenan modelos usando tanto métodos de [AA](#) supervisados como basándose en reglas.

**TOKENIZADOR:** *Tokeniza* el texto en secuencias de símbolos. El componente para el Inglés proporciona un *tokenizador* al estilo *Penn Tree Bank* (PTB) que ha sido ampliado para tratar con texto de la web y con ruido. Los componentes correspondientes para el Chino y Árabe proporcionan segmentación de palabras y *clitic*. Como proceso final, el *tokenizador* almacena los desplazamientos de caracteres de cada símbolo en el texto de entrada.

**CLEANXML:** Elimina las etiquetas *Extensible Markup Language* (XML) del documento.

**SSPLIT:** Separa una secuencia de símbolos en frases.

**TRUECASE:** Determina la probabilidad de un símbolo de estar en mayúscula en el texto — es decir, la probabilidad de que, en un texto bien escrito, dicho símbolo debiera estar en mayúscula. — cuando esta información no está disponible, por ejemplo, un texto con todas las letras en mayúsculas. Este proceso se implementa con un modelo discriminativo usando un etiquetador de secuencias [CRF](#) [5].

**POS:** Etiqueta símbolos con su correspondiente categoría morfosintáctica — *POS tag* — haciendo uso de un etiquetador de máxima entropía [22].

**LEMMA:** Genera la raíz — o *lemma* — para todos los símbolos en la anotación.

**GENDER:** Añade información sobre el género más probable a los nombres.

**NER:** Reconoce nombres — PERSONA, LUGAR, ORGANIZACIÓN, MISCELÁNEA — y entidades numéricas — DINERO, NÚMERO, FECHA, HORA, DURACIÓN. — En los etiquetados por defecto, las entidades para nombres se reconocen mediante una combinación de secuencias de [CRFs](#) entrenados en varios *corpus* [5]. Las entidades numéricas son re-

conocidas usando dos sistemas basados en reglas, uno para el dinero y números, y otro sistema *estado del arte* para procesar expresiones temporales [3].

**REGEXNER:** Implementa un NER simple basado en reglas usando secuencias de símbolos mediante expresiones regulares. El objetivo de este paso del *pipeline* es proporcionar un *framework* que permita al usuario incorporar etiquetas NE que no están anotadas en *corpus* NL tradicionales. Por ejemplo, la lista por defecto de expresiones regulares distribuida en los modelos de CORENLP reconoce ideologías, nacionalidades, religiones y títulos.

**PARSE:** Proporciona un análisis sintáctico completo, incluyendo representaciones tanto de dependencias como constituyentes. Está basado en un parseador probabilístico [9, 15].

**SENTIMENT:** Análisis de sentimientos con un modelo compositivo sobre árboles usando *deep learning* [20]. La puntuación asignada a un sentimiento se calcula mediante los nodos de un árbol binario para cada sentencia, incluyendo, en particular, el nodo raíz de cada frase.

**DCOREF:** Detecta menciones y resolución de coreferencias tanto pronominales como nominales [11]. Se devuelve el grafo de coreferencia del texto al completo, con las palabras principales de las menciones como nodos.

En la [Figura 3](#) se muestra el *pipeline* completo de CORENLP.

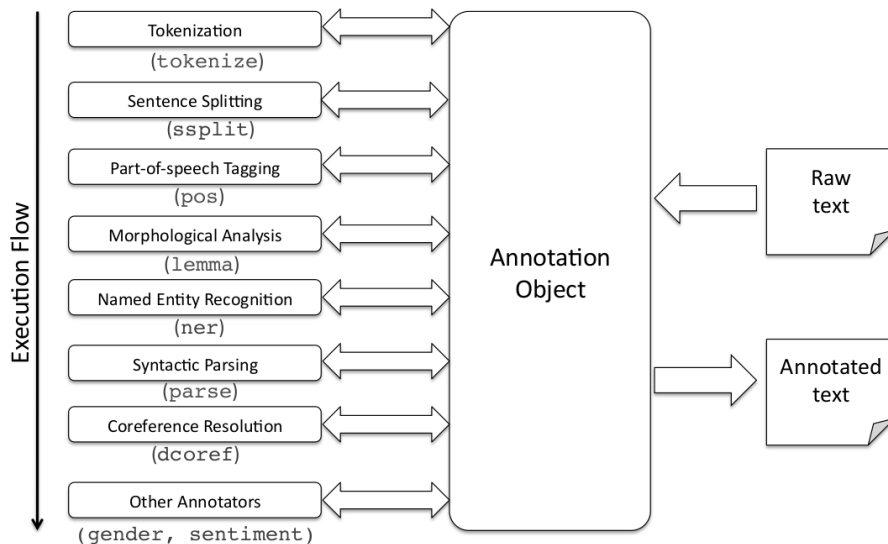


Figura 3: Visión general de la arquitectura. El texto a procesar se añade a un objeto `ANNOTATION`. Posteriormente una secuencia de etiquetadores añaden información sobre qué fases del *pipeline* deben ejecutarse. El resultado contiene el texto procesado, puede obtener en formato [XML](#) o texto plano.

## 1.6 SOFTWARE EXISTENTE

Menciona software existente?

## 1.7 ESTADO DEL ARTE

En esta sección se describirán las aproximaciones actuales en el análisis de sentimientos, en función de los distintos niveles mencionados en [Niveles de análisis](#).

## 1.7.1 A nivel del documento

Semantic  
Orientation (SO):  
Orientación  
Semántica

Turney [25] presentó un método no supervisado para clasificar reseñas como recomendadas o no recomendadas. Su investigación determina las polaridades mediante una media de la *Semantic Orientation* (SO) de las frases que aparecen en las reseñas. La SO de un par de frases se calcula mediante *Pointwise Mutual Information and Information Retrieval* (PMI-IR) [24], usando la [Ecuación 1](#)

$$SO(\text{frase}) = \log \frac{\text{hits}(\text{frase}, \text{"excelente"}) \text{hits}(\text{"mediocre"})}{\text{hits}(\text{frase}, \text{"mediocre"}) \text{hits}(\text{"excelente"})} \quad (1)$$

Pang, Lee y Vaithyanathan [16] emplearon métodos de AA para clasificar reseñas como positivas o negativas. Para ello entrenaron los siguientes clasificadores: naïve Bayes, máxima entropía y *Support Vector Machine* (SVM) usando las siguientes características: unigrama, bigrama, POS tag y la posición de la palabra. Los autores indicaron que las ocurrencias de opiniones contradictorias hacen del análisis de sentimientos para documentos una tarea más difícil que la categorización de texto basada en temas – *topic-based*. –

## 1.7.2 A nivel de sentencia

Yu y Hatzivassiloglou [29] presentaron tres aproximaciones para clasificar la subjetividad, basada en la similitud de sentencias, naïve Bayes y varios clasificadores naïve Bayes, respectivamente. La primera aproximación, basada en similitud, calcula una puntuación de similitud de las frases y las compara con un umbral predeterminado. La aproximación usando naïve Bayes entrena un clasificador con las sentencias de documentos de opinión y fácticos. Las características son unigramas, bigramas, trigramas POS tags y un recuento de palabras expresando opinión. Los múltiples clasificadores naïve Bayes entrena a los distintos clasificadores en diferentes sub conjuntos del espacio de características. Posteriormente las sentencias que tienen etiquetas diferentes con los correspondientes documentos se eliminan del conjunto de test. El proceso de entrenamiento y testeo es iterativo hasta

que no haya más frases en el conjunto de test a borrar. Finalmente, los clasificadores entrenados en el conjunto de entrenamiento reducido son los usados para determinar si las sentencias son subjetivas u objetivas.

### 1.7.3 *A nivel de entidad*

Thet, Na y Khoo [21] propusieron una aproximación lingüística aprovechando la estructura de dependencia gramatical de las cláusulas. La estructura de la dependencia gramatical se obtiene a través de un árbol sintáctico, para luego extraer sub-árboles que representan cláusulas — por ejemplo, dividiendo una frase en oraciones separadas que expresan opiniones hacia el aspecto correspondiente. — La generación del sub-árbol puede considerarse como una descomposición de una sentencia compleja en varias simples, transformando el problema original a una tarea a nivel de sentencias. La extracción del aspecto en este trabajo se simplifica a un proceso de etiquetado semántico. En este proceso se asigna una etiqueta a cada oración mediante un conjunto de aspectos predefinidos indicando palabras.

Wang y col. [28] usaron un modelo no supervisado, basado en máquinas de Boltzmann restringidas. El modelo propuesto extrae conjuntamente sentimiento y aspecto. Además el modelo incorpora como variables latentes las variables aspecto, sentimiento y demás variables de fondo. Sin embargo, estas variables latentes no poseen una relación condicional, es decir, no hay aristas conectándolas en el modelo gráfico.





## Parte II

### OBJETIVOS

En este apartado expondremos la necesidad de desarrollar un software de parseo de dependencias para el idioma Español. Se revisará el estado del arte para este proceso del *pipeline* y por último, se introducirá la propuesta para resolver el problema.

---

En este apartado deberán aparecer con claridad los objetivos, inicialmente previstos en la propuesta de TFG, indicando el alcance para cada uno de ellos. Es conveniente indicar de manera precisa las interdependencias entre los distintos objetivos y también enlazarlos con los diferentes apartados de la memoria.

Los aspectos formativos previos más utilizados pueden ser destacados aquí.



Como se comentó en la [Sección 1.2](#), el [DP](#) establece relaciones entre palabras principales y sus modificadores. Sin embargo, en la [Sección 1.3](#) se mencionó la escasez de software que realice este tipo de tareas para idiomas distintos del Inglés. Este hecho justifica el desarrollo de un parseador de dependencias para el idioma Español. En este capítulo se revisarán los métodos existentes para tal fin, así como el desarrollado en el trabajo.

Comenzaremos con el trabajo de Ballesteros y col. [[2](#)]



## Parte III

### RESOLUCIÓN DEL TRABAJO

Se explicarán los métodos y procesos empleados para desarrollar el trabajo y alcanzar los objetivos. Es conveniente destacar tanto los métodos inicialmente previstos como aquellos que hayan tenido que ser agregados en el desarrollo del trabajo.

Éste es el lugar de presentar todos los datos técnicos y científicos realizados en el TFG. Debe ser detallado, claro y preciso.

En caso de ser un TFG en el que se desarrolle software, se recomienda que la sección 5 quede estructurada de la siguiente forma:

5.1 Planificación y presupuesto. Planificación temporal, con su correspondiente división en fases y tareas, y la posterior comparación con los datos reales obtenidos tras realizar el proyecto. También se incluir un presupuesto del trabajo a realizar.

5.2 Análisis y diseño. Se incluirá la especificación de requerimientos y la metodología de desarrollo por la que se ha optado, así como los “planos” del proyecto, que contendrán las historias de usuario o casos de uso, diagrama conceptual, diagramas de iteración, diagramas de diseño, esquema arquitectónico y bocetos de las interfaces de usuario. Además, se describirán las estructuras de datos fundamentales y los desarrollos algorítmicos no triviales.

5.3. Implementación y pruebas. En esta sección se incluirán todos los aspectos relacionados con la programación de la aplicación y las tecnologías seleccionadas, justificándolas e incluyendo el diseño de pruebas e informes de ejecución de las mismas.



## Parte IV

### CONCLUSIONES Y VÍAS FUTURAS

Las conclusiones deben incluir todas aquellas de tipo profesional y académico. Además, se debe indicar si los objetivos han sido alcanzados totalmente, parcialmente o no alcanzados.

Si hubiese claramente posibles vías de desarrollo posterior es interesante destacarlas, poniéndolas en valor en el contexto inicial del trabajo.





Parte V

APPENDIX



## BIBLIOGRAFÍA

---

- [1] Michael Gamon Anthony Aue. «Customizing Sentiment Classifiers to New Domains: a Case Study». En: *Submitted to RANLP-05, the International Conference on Recent Advances in Natural Language Processing*. Borovets, BG, 2005.
- [2] Miguel Ballesteros, Bernd Bohnet, Simon Mille y Leo Wanner. «Data-driven deep-syntactic dependency parsing». En: *Natural Language Engineering* 22.6 (nov. de 2016), págs. 939-974.
- [3] Angel X. Chang y Christopher D. Manning. «SUTIME: A Library for Recognizing and Normalizing Time Expressions». En: *In LREC*. 2012.
- [4] Ronen Feldman. «Techniques and applications for sentiment analysis». En: *Communications of the ACM* 56.4 (2013), pág. 82.
- [5] Jenny Rose Finkel, Trond Grenager y Christopher Manning. «Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling». En: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. ACL '05. Ann Arbor, Michigan: Association for Computational Linguistics, 2005, págs. 363-370.
- [6] Geoffrey Hinton y col. «Deep Neural Networks for Acoustic Modeling in Speech Recognition». En: *IEEE Signal Processing Magazine* 29 (2012), págs. 82-97.
- [7] J. Hirschberg y C. D. Manning. «Advances in natural language processing». En: *Science* 349.6245 (2015), págs. 261-266.
- [8] Minqing Hu y Bing Liu. «Mining and summarizing customer reviews». En: *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*. Association for Computing Machinery (ACM), 2004.
- [9] Dan Klein y Christopher D. Manning. «Fast Exact Inference with a Factored Model for Natural Language Parsing». En: *Proceedings of the 15th International Conference on Neural Information Processing Systems*. NIPS'02. Cambridge, MA, USA: MIT Press, 2002, págs. 3-10.
- [10] Taku Kudo, Kaoru Yamamoto y Yuji Matsumoto. «Applying conditional random fields to Japanese morphological analysis». En: *In Proc. of EMNLP*. 2004, págs. 230-237.

- [11] Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu y Dan Jurafsky. «Deterministic Coreference Resolution Based on Entity-centric, Precision-ranked Rules». En: *Comput. Linguist.* 39.4 (dic. de 2013), págs. 885-916. ISSN: 0891-2017.
- [12] B. Liu. En: *Handbook of Natural Language Processing Chapter Sentiment Analysis and Subjectivity* (2010), págs. 627-666.
- [13] Bing Liu. «Sentiment Analysis and Opinion Mining». En: *Synthesis Lectures on Human Language Technologies* 5.1 (2012), págs. 1-167.
- [14] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard y David McClosky. «The Stanford CoreNLP Natural Language Processing Toolkit». En: *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics (ACL), 2014.
- [15] Marie-Catherine de Marneffe y Christopher D. Manning. «The Stanford Typed Dependencies Representation». En: *Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*. CrossParser '08. Manchester, United Kingdom: Association for Computational Linguistics, 2008, págs. 1-8. ISBN: 978-1-905593-50-7.
- [16] Bo Pang, Lillian Lee y Shivakumar Vaithyanathan. «Thumbs Up?: Sentiment Classification Using Machine Learning Techniques». En: *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*. EMNLP '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, págs. 79-86.
- [17] Fuchun Peng, Fangfang Feng y Andrew McCallum. «Chinese segmentation and new word detection using conditional random fields». En: *Proceedings of the 20th international conference on Computational Linguistics - COLING '04*. Association for Computational Linguistics (ACL), 2004.
- [18] Mike Pound y Sean Riley. *Inside a Neural Network – Computerp-hile*. Youtube. 2016. URL: [https://www.youtube.com/watch?v=BFdMrD0x\\_CM](https://www.youtube.com/watch?v=BFdMrD0x_CM).
- [19] James Pustejovsky y Amber Stubbs.
- [20] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng y Christopher Potts. «Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank». En: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, 2013, págs. 1631-1642.

- [21] Tun Thura Thet, J.-C. Na y C. S. G. Khoo. «Aspect-based sentiment analysis of movie reviews on discussion boards». En: *Journal of Information Science* 36.6 (2010), págs. 823-848.
- [22] Kristina Toutanova, Dan Klein, Christopher D. Manning y Yo-ram Singer. «Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network». En: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*. NAACL '03. Edmonton, Canada: Association for Computational Linguistics, 2003, págs. 173-180.
- [23] Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky y Christopher Manning. «A Conditional Random Field Word Segmenter for Sighan Bakeoff 2005». En: *SIGHAN Workshop on Chinese Language Processing*. Association for Computational Linguistics, 2005.
- [24] Peter D. Turney. «Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL». En: *Machine Learning: ECML 2001: 12th European Conference on Machine Learning Freiburg, Germany, September 5-7, 2001 Proceedings*. Ed. por Luc De Raedt y Peter Flach. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, págs. 491-502. ISBN: 978-3-540-44795-5.
- [25] Peter D. Turney. «Thumbs Up or Thumbs Down?: Semantic Orientation Applied to Unsupervised Classification of Reviews». En: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. ACL '02. Philadelphia, Pennsylvania: Association for Computational Linguistics, 2002, págs. 417-424.
- [26] David Vilares, Miguel A. Alonso y Carlos Gómez-Rodríguez. «A syntactic approach for opinion mining on Spanish reviews». En: *Natural Language Engineering* 21.01 (2013), págs. 139-163.
- [27] Hao Wang y Martin Ester. «A Sentiment-aligned Topic Model for Product Aspect Rating Prediction». En: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics (ACL), 2014.
- [28] Linlin Wang, Kang Liu, Zhu Cao, Jun Zhao y Gerard de Melo. «Sentiment-Aspect Extraction based on Restricted Boltzmann Machines». En: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, 2015, págs. 616-625.
- [29] Hong Yu y Vasileios Hatzivassiloglou. «Towards Answering Opinion Questions: Separating Facts from Opinions and Identifying the Polarity of Opinion Sentences». En: *Proceedings of*

- the 2003 Conference on Empirical Methods in Natural Language Processing*. EMNLP '03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, págs. 129-136.
- [30] Lina Zhou y Pimwadee Chaovalit. «Ontology-supported polarity mining». En: *Journal of the American Society for Information Science and Technology* 59.1 (2007), págs. 98-110.

## DECLARATION

---

Put your declaration here.

*Granada, 23 de noviembre de 2016*

---

Alejandro Alcalde Barros





## COLOPHON

This document was typeset using the typographical look-and-feel *classicthesis* developed by André Miede. The style was inspired by Robert Bringhurst’s seminal book on typography “*The Elements of Typographic Style*”. *classicthesis* is available for both  $\text{\LaTeX}$  and  $\text{\LyX}$ :

<https://bitbucket.org/amiede/classicthesis/>

Happy users of *classicthesis* usually send a real postcard to the author, a collection of postcards received so far is featured here:

<http://postcards.miede.de/>

Hermann Zapf’s *Palatino* and *Euler* type faces (Type 1 PostScript fonts *URW Palladio L* and *FPL*) are used. The “typewriter” text is typeset in *Bera Mono*, originally developed by Bitstream, Inc. as “Bitstream Vera”. (Type 1 PostScript fonts were made available by Malte Rosenau and Ulrich Dirr.)

NOTE: The custom size of the textblock was calculated using the directions given by Mr. Bringhurst (pages 26–29 and 175/176). 10 pt *Palatino* needs 133.21 pt for the string “abcdefghijklmnopqrstuvwxyz”. This yields a good line length between 24–26 pc (288–312 pt). Using a “*double square textblock*” with a 1:2 ratio this results in a textblock of 312:624 pt (which includes the headline in this design). A good alternative would be the “*golden section textblock*” with a ratio of 1:1.62, here 312:505.44 pt. For comparison, `DIV9` of the `typearea` package results in a line length of 389 pt (32.4 pc), which is by far too long. However, this information will only be of interest for hardcore pseudo-typographers like me.

To make your own calculations, use the following commands and look up the corresponding lengths in the book:

```
\settowidth{\abcd}{abcdefghijklmnopqrstuvwxyz}
\the\abcd\ % prints the value of the length
```

Please see the file `classicthesis.sty` for some precalculated values for *Palatino* and *Minion*.

145.86469pt