UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

IDENTIFICACIÓN DE LA PRESENCIA DE IRONÍA EN EL TEXTO GENERADO POR USUARIOS DE TWITTER UTILIZANDO TÉCNICAS DE OPINION MINING Y MACHINE LEARNING

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

VÍCTOR ALEJANDRO HERNÁNDEZ MARTÍNEZ

PROFESOR GUÍA:
SR. JUAN DOMINGO VELÁSQUEZ SILVA

MIEMBROS DE LA COMISIÓN:
SR. FRANCISCO MOLINA JARA
SR. ALBERTO CABEZAS BULLEMORE

SANTIAGO DE CHILE
MAYO 2015

Resumen Ejecutivo

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL INDUSTRIAL
POR : VÍCTOR HERNÁNDEZ MARTÍNEZ
FECHA: 25/05/2015
PROF. GUIA: SR. JUAN VELÁSQUEZ

El siguiente trabajo tiene como objetivo general diseñar e implementar un módulo clasificador de texto que permita identificar la presencia de ironía en el contenido generado por usuarios de Twitter, mediante el uso de herramientas asociadas a Opinion Mining y Machine Learning. La ironía es un fenómeno que forma parte del contenido generado por las personas en la Web, y representa un campo de estudio nuevo que ha atraído la atención de algunos investigadores del área de Opinion Mining debido a su complejidad y al impacto que puede tener en el desempeño de las aplicaciones de Análisis de Sentimientos actuales. Este trabajo de título se desarrolla dentro del marco de OpinionZoom, proyecto CORFO código 13IDL2-23170 titulado *"OpinionZoom: Plataforma de análisis de sentimientos e ironía a partir de la información textual en redes sociales para la caracterización de la demanda de productos y servicios"* desarrollado en el Web Intelligence Centre del Departamento de Ingeniería Industrial de la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile, el cual busca generar un sistema avanzado para analizar datos extraídos desde redes sociales para obtener información relevante para las empresas en relación a sus productos y servicios.

La hipótesis de investigación de este trabajo dice que es posible detectar la presencia de ironía en texto en idioma Español con cierto nivel de precisión, utilizando una adaptación de la metodología propuesta por Reyes et al. (2013) en [5] la cual involucra la construcción de un corpus en función de la estructura de Twitter junto con la capacidad de las personas para detectar ironía.

El modelo utilizado se compone de 11 atributos entre los cuales se rescatan características sintácticas, semánticas y emocionales o psicológicas, con el objetivo de poder describir ironía en texto. Para esto, se genera un corpus de casos irónicos y no irónicos a partir de una selección semiautomática utilizando una serie de hashtags en Twitter, para luego validar su etiquetado utilizando evaluadores humanos. Además, esto se complementa con la inclusión de textos objetivos como parte del set de casos no irónicos. Luego, utilizando este corpus, se pretende realizar el entrenamiento de un algoritmo de aprendizaje supervisado para realizar la posterior clasificación de texto. Para ésto, se implementa un módulo de extracción de atributos que transforma cada texto en un vector representativo de los atributo. Finalmente, se utilizan los vectores obtenidos para implementar un módulo clasificador de texto, el cual permite realizar una clasificación entre tipos irónicos y no irónicos de texto. Para probar su desempeño, se realizan dos pruebas. La primera utiliza como casos no irónicos los textos objetivos y la segunda utiliza como casos no irónicos aquellos textos evaluados por personas como tales. La primera obtuvo un alto nivel de precisión, mientras que la segunda fue insuficiente. En base a los resultados se concluye que esta implementación no es una solución absoluta. Existen algunas limitaciones asociadas a la construcción del corpus, las herramientas utilizadas e incluso el modelo, sin embargo, los resultados muestran que bajo ciertos escenarios de comparación, es posible detectar ironía en texto por lo que se cumple la hipótesis. Se sugiere ampliar la investigación, mejorar la obtención del corpus, utilizar herramientas más desarrolladas y analizar aquellos elementos que el modelo no puede capturar.

Strength does not come from physical capacity. It comes from an indomitable will.
- Mahatma Gandhi

Agradecimientos

En primer lugar quiero agradecer y dedicar este trabajo a mi familia, principalmente a mis padres, mis hermanos y mis abuelos, quienes siempre han estado a mi lado y siempre me han dado apoyo cuando lo he necesitado. Todo lo que soy en estos momentos se lo debo a ellos, por sus enseñanzas, sus alientos, sus cuidados y por el solo hecho de estar presentes en todo instante. No existen palabras que me permitan describir mi gratitud, y me considero un hombre muy afortunado por tener una familia como ustedes.

Agradezco al profesor Juan Velásquez por haber depositado su confianza en mí, por entregarme este y otros desafíos, por su comprensión y por haberme dado tantas oportunidades a lo largo de mi vida universitaria. También a Francisco Molina, por sus buenos consejos, su apoyo y comprensión a lo largo del desarrollo de mi trabajo. Estoy muy agradecido y contento de haber formado parte de este proceso. Además, le agradezco a todos los integrantes del WIC que conocí durante el desarrollo de este trabajo y con quienes pasaba gran parte del día en *La Salita*, por toda la ayuda y el apoyo recibido.

También quiero agradecer a mis amigos que conocí durante estos años en la universidad. Chino, Seba Curinao, Fano, Negro, Rolo, Willy, Nacho, Pablo Jerez, Pablo Zamorano, Cabezón, Lucho, Icho, Pasita, Gilla, Coni, Kol, Gabiluli, Mila y Nico Pacheco. Gracias a todos ustedes por los buenos momentos vividos, los terraceos, las pichangas, los estudios en eléctrica, geología y luego en el DII, los almuerzos en los pastos casi todos los días, los carretes, las sonrisas espontáneas y el cariño recibido. En verdad los estimo mucho a todos, son muy importantes para mí. Los quiero mucho y espero que esto no se pierda en el camino nunca.

Además quiero agradecer a toda la gente que conocí en el equipo de la Feria Empresarial de la facultad, entre el 2011 y el 2014. Aprendí mucho de ellos y de ahí aun mantengo grandes lazos de amistad. Gracias por todo, fue un agrado haber pertenecido al proyecto durante tantos años. También quiero agradecer a la gente que conocí en los Equipazos CEIN 2013 y 2014, quienes siempre fueron muy inclusivos con los alumnos y siempre me dieron espacio para participar. En particular, le agradezco a Rita y Cristian, quienes siempre me invitaban a participar de los proyectos y que además hicieron mucho por mí.

A todos ustedes, Gracias Totales!

Víctor Alejandro Hernández Martínez

Tabla de Contenido

Resumen Ejecutivo **i**

Agradecimientos **iii**

Índice de Tablas **vii**

Índice de Figuras **viii**

1 Introducción **1**
1.1 Antecedentes Generales . . . 1
1.2 Descripción del proyecto y justificación . . . 2
1.3 Objetivos . . . 3
1.3.1 Objetivo General . . . 3
1.3.2 Objetivos Específicos . . . 3
1.4 Hipótesis de Investigación . . . 4
1.5 Metodología . . . 4
1.6 Resultados Esperados y Alcances . . . 5
1.7 Estructura del Informe . . . 6

2 Marco Conceptual **7**
2.1 La Web y el contenido generado por sus usuarios . . . 7
2.1.1 World Wide Web en sus inicios . . . 7
2.1.2 Web 2.0 . . . 8
2.1.3 Principios fundamentales de la Web 2.0 . . . 9
2.2 Knowledge Discovery y Data Mining . . . 10
2.2.1 Knowledge Discovery in Databases . . . 10
2.2.2 Data Mining . . . 12
2.3 Web Mining . . . 13
2.3.1 Definición . . . 13
2.3.2 Categorías . . . 13
2.3.3 Algunas aplicaciones . . . 14
2.4 Web Opinion Mining . . . 14
2.4.1 Opinion Mining o Sentiment Analysis y su potencial . . . 14
2.4.2 Aplicaciones . . . 15
2.4.3 Enfoques de Análisis . . . 16
2.4.4 El problema de la Ironía . . . 17
2.5 Machine Learning . . . 18
2.5.1 Definición . . . 18
2.5.2 Aprendizaje Supervisado . . . 19
2.5.3 Text Mining . . . 20
2.5.4 Pre-procesamiento de texto . . . 21
2.5.5 Extracción de Atributos . . . 22
2.5.6 Métodos de Aprendizaje Supervisado . . . 23
2.6 Twitter . . . 25
2.6.1 Twitter y el concepto de Microblogging . . . 25
2.6.2 Usuarios . . . 26

2.6.3 Contenido . . . 26
2.6.4 Trending Topics . . . 27
2.7 Ironía . . . 27
2.7.1 Lenguaje Figurado . . . 27
2.7.2 Definición de Ironía . . . 28
2.7.3 Rol en la comunicación . . . 29
2.8 Detección de ironía presente en texto . . . 30
2.8.1 Procedimientos Actuales . . . 31
2.9 Análisis y Evaluación de Resultados . . . 32
2.9.1 Métricas de Desempeño . . . 32
2.9.2 K-fold Cross Validation . . . 34

3 Construcción del corpus **35**
3.1 Definición . . . 35
3.2 Corpus para la detección de ironía . . . 35
3.2.1 Hashtags . . . 36
3.2.2 Uso de Hashtags para recolectar datos . . . 37
3.3 Método de Extracción . . . 38
3.3.1 Fuentes Utilizadas . . . 38
3.3.2 Módulo de Extracción de Datos . . . 38
3.4 Análisis Exploratorio y Limpieza de Datos . . . 39
3.4.1 Problemas identificados en los datos extraídos . . . 39
3.4.2 Solución propuesta a problemas identificados . . . 41
3.5 Clasificación Manual . . . 42
3.5.1 Primera Evaluación . . . 43
3.5.2 Segunda Evaluación . . . 45
3.6 Sets de Casos Negativos . . . 45

4 Modelo de detección de Ironía y Método de Clasificación **47**
4.1 Modelo de Detección de Ironía . . . 47
4.1.1 Dimensiones . . . 47
4.2 Método de Clasificación . . . 52
4.2.1 Clasificador Naive Bayes . . . 52

5 Implementación **54**
5.1 Herramientas Tecnológicas . . . 54
5.2 Organización y Evaluación de Tweets . . . 55
5.3 Módulo de Extracción de Atributos . . . 57
5.3.1 Modelo de Datos . . . 57
5.3.2 Pre-procesamiento Básico . . . 59
5.3.3 Signatures . . . 60
5.3.4 Unexpectedness . . . 62
5.3.5 Style . . . 66
5.3.6 Emotional Contexts . . . 68
5.4 Módulo de Clasificación . . . 70
5.4.1 Entrenamiento, Pruebas y Evaluación . . . 71

6 Resultados **73**
6.1 Extracción de Atributos . . . 73
6.2 Clasificación . . . 79
6.3 Discusión . . . 83
6.3.1 Construcción del Corpus . . . 83
6.3.2 Modelo y Herramientas utilizadas . . . 85

7 Conclusiones **87**
7.1 Trabajo Futuro 90

Referencias **91**

Apéndices **96**
A Utilidades de Implementación 96
A .1 Pre-procesamiento 96
A .2 Implementación Dimensión Signatures 97

Índice de Tablas

2.1 Ejemplo de Matriz de Confusión. 33

3.1 Resultados de filtrado automático de Tweets. 42
3.2 Resultados de evaluación inicial de tweets irónicos. 43
3.3 Ejemplo de Matriz de proporciones utilizada para obtener el kappa de Cohen de un set. 43
3.4 Cálculo de medida kappa para cada set. 44
3.5 Resultados de evaluación de discrepancias en la evaluación inicial. 45
3.6 Resultado final de evaluación manual de tweets. 45
3.7 Cuentas de prensa utilizadas para la recolección de texto objetivo. 46

4.1 Dimensiones del Modelo de Detección de Ironía junto a sus atributos. 48

5.1 Ejemplo de representación Bag-of-Words para un conjunto de documentos. . 67
5.2 Datos de los recursos utilizados para el etiquetado de polaridad. 68
5.3 Cantidad de datos positivos y negativos distribuidos para las pruebas. 70

6.1 Detección de patrones del atributo *Pointedness*. 74
6.2 Detección de patrones de los atributos *Counterfactual* y *Temporal Compression*. 74
6.3 Emoticonos más frecuentes detectados en el atributo *Pointedness*. 74
6.4 Palabras más frecuentes detectadas en los atributos *Counterfactual* y *Temporal Compression*. 75
6.5 Detección de patrones del atributo *Contextual Imbalance*. 75
6.6 Detección de patrones del atributo *Temporal Imbalance*. 75
6.7 Detección de patrones del atributo *Activation*. 78
6.8 Detección de patrones del atributo *Imagery*. 79
6.9 Detección de patrones del atributo *Pleasantness*. 79
6.10 Niveles de Accuracy obtenidos al agregar atributos en forma progresiva para ambas pruebas 79
6.11 Resultados finales de Prueba 1 81
6.12 Resultados de 10-Fold Cross Validation de Prueba 1 81
6.13 Resultados finales de Prueba 2 82
6.14 Resultados de 10-Fold Cross Validation de Prueba 2 82
6.15 Resultados de Prueba 1 desglosados en una Matriz de Confusión. 83
6.16 Resultados de Prueba 2 desglosados en una Matriz de Confusión. 83

Índice de Figuras

2.1 Proceso KDD 10
2.2 Proceso de Aprendizaje Supervisado 20
2.3 Ejemplo de POS-Tagging de una oración 23
2.4 Ejemplo de un Árbol de Decisión para la predicción de clientes compradores de computador 24
2.5 Hiperplano de margen máximo en método SVM 25
2.6 Ejemplo de 10-fold Cross Validation, uno de los casos más utilizados. 34

5.1 Tabla *tweet* donde se almacenan datos asociados a los tweets recolectados para la clasificación 57
5.2 Modelo de datos utilizado en la extracción de atributos 58
5.3 Esquema de Organización y Evaluación de Tweets 59
5.4 Esquema dimensión Signatures 61
5.5 Esquema dimensión Unexpectedness 64
5.6 Esquema dimensiones Style y Emotional Contexts 70

6.1 Frecuencia de Polarity Skipgrams encontrados en Pruebas 1 y 2 76
6.2 50 cgrams más frecuentes encontrados en Prueba 1 76
6.3 50 cgrams más frecuentes encontrados en Prueba 2 77
6.4 50 Skipgrams más frecuentes encontrados en Prueba 1 77
6.5 50 Skipgrams más frecuentes encontrados en Prueba 2 78
6.6 Gráfico de niveles de Accuracy obtenidos al agregar atributos en forma progresiva para ambas pruebas 80
6.7 Gráfico de niveles de Accuracy obtenidos al agregar atributos en forma progresiva para Prueba 2 80

7.1 Expresión Regular para Tokenización 96
7.2 Lista de Emoticonos para Pre-procesamiento 97

Capítulo 1

Introducción

El siguiente capítulo da una introducción al proyecto de investigación aplicada realizado. En primer lugar se dan a conocer algunos antecedentes generales que rodean a la problemática atendida, para luego describir a grandes rasgos el proyecto junto a su justificación. Se da a conocer el objetivo general del trabajo y los objetivos específicos requeridos para alcanzarlo. Se plantea una hipótesis de investigación para el proyecto y una metodología, seguida de los resultados que se espera obtener y los alcances que tendrá el proyecto. Finalmente, se describe la estructura que conforma este informe.

1.1 Antecedentes Generales

En la actualidad, Internet es uno de los medios más utilizados en el mundo. La cantidad de usuarios con acceso a éste ha aumentado desde un 1% de la población mundial en 1995 hasta un 40% aproximadamente en el 2014, acercándose al trillón de usuarios [1]. En particular, Chile ha participado de este crecimiento en el tiempo y actualmente un 60% de sus habitantes son considerados usuarios de Internet. Con la existencia de la Web 2.0, los usuarios tienen a su alcance diversos contenidos pero, además son capaces de generar contenido propio, el cual pueden compartir con el resto del mundo. Este crecimiento ha provocado que la cantidad y diversidad de la información presente en la Web sea inmensa, lo que ha captado interés de las masas respecto a la hora de tomar una decisión.

Dentro del contenido generado por los usuarios, es posible encontrar análisis y opiniones de diversos productos en grandes cantidades y realizados por distintos usuarios a través de distintos medios como blogs, foros de opinión, redes sociales, sitios especializados, etc. Los consumidores actuales ya no están limitados a informarse solo con publicidad o con la percepción de otras personas cercanas respecto a un producto en particular para poder tomar su decisión de compra. Estas fuentes no solo son útiles para los potenciales consumidores si no que también para las empresas, las cuales están tendiendo a utilizar la información que estos medios les entregan, sin depender exclusivamente de otras herramientas de estudio como encuestas, entrevistas, focus groups, etc. Sin embargo, la creación de sistemas avanzados que se ocupen de administrar esta enorme y creciente cantidad de datos para poder extraer

información relevante sigue siendo un desafío.

La minería de opiniones corresponde al tratamiento computacional de opiniones, sentimientos y subjetividad presentes en texto. Durante la década del 2000, este campo se transformó en un área bastante estudiada debido a diversos motivos. Uno de ellos, es que involucra una amplia gama de aplicaciones en diversos dominios, entre estos, aplicaciones de inteligencia de negocios. Lamentablemente, a pesar de que el área tiene aplicaciones de interés para diversas entidades presentes en la actualidad, también involucra una serie de limitantes asociadas a la naturaleza propia del texto que los usuarios generan. Una de éstas es la presencia de ironía, la cual es utilizada por los usuarios web inmersos en contextos sociales, los cuales representan parte de las principales fuentes de contenido relevante para el estudio.

1.2 Descripción del proyecto y justificación

A raíz del explosivo crecimiento en el tiempo del contenido generado por los usuarios de la Web 2.0, tanto consumidores como empresas tienen acceso a abundantes cantidades de datos de diversa especie. El contenido relacionado con opiniones sobre distintos productos y servicios tiene una gran presencia en la Web y su importancia radica en que además impacta en la decisión de compra de los distintos consumidores que utilizan estos medios para informarse [2]. Ante esta evidencia, cada vez más empresas se interesan en el campo del *Business Intelligence* específicamente aplicado al contenido web con el objetivo de extraer información relevante de éste. Esta aplicación deriva en lo que se conoce como *Web Intelligence*, campo definido como el uso de técnicas avanzadas de Inteligencia Artificial y Tecnologías de Información para el propósito de explorar, analizar y extraer conocimiento del contenido web [3]. El buen uso de estas técnicas resulta interesante para las empresas ya que representan métodos alternativos de investigación de mercado, de gran utilidad y benéficos en comparación a otros métodos existentes que demandan una alta cantidad de recursos y tiempo (*Focus Groups*, Encuestas, Entrevistas, etc.).

En función de esta necesidad, surge el interés en desarrollar sistemas avanzados que permitan extraer información de este contenido en forma automática. Uno de los caminos para esto es utilizar técnicas de Opinion Mining o Análisis de Sentimientos, lo que puede definirse como el tratamiento computacional de opiniones, sentimientos y otros elementos subjetivos presentes en texto [4]. En la actualidad, el uso de estas técnicas tiene limitaciones asociadas a las características propias del contenido que los usuarios de la Web 2.0 generan en forma espontánea en los distintos medios sociales que manejan habitualmente.

El siguiente proyecto gira en torno a una de las limitaciones que se identifican en la literatura actual: La presencia de ironía en el contenido generado por usuarios de la Web 2.0. Las fuentes utilizadas para obtener este contenido se encuentran generalmente dentro un contexto social donde los distintos usuarios interactúan unos con otros, y el texto presente en estos medios suele contener ironía [5].

Por lo tanto, este proyecto apunta a culminar en el diseño de un módulo clasificador de ironía en texto, el cual utilizando técnicas conocidas de análisis de texto y Opinion Mining,

permita identificar con cierto grado de verosimilitud, opiniones y/o contenido generado por usuarios de la Web 2.0 construido con texto irónico o no irónico. ¿Por qué se considera que la ironía representa una limitación? Generalmente, al utilizar técnicas de Opinion Mining o Análisis de Sentimientos, se busca identificar y clasificar la polaridad que caracteriza el texto analizado y esto puede ser "positivo" o "negativo" [6]. Sin embargo, la presencia de ironía entorpece esta clasificación ya que un texto con ironía contiene un significado implícito, no presente en el significado textual de la expresión [7]. Por lo tanto, se considera que al generar métodos que puedan detectar la presencia de ironía en el texto, es posible alcanzar mejoras sustanciales en el desempeño de las metodologías clásicas de la Opinion Mining.

Para finalizar con la contextualización, cabe mencionar que el siguiente proyecto se desarrolla dentro del marco de OpinionZoom, proyecto CORFO código 13IDL2-23170 titulado "OpinionZoom: Plataforma de análisis de sentimientos e ironía a partir de la información textual en redes sociales para la caracterización de la demanda de productos y servicios" desarrollado en el Web Intelligence Centre[1] del Departamento de Ingeniería Industrial de la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile, el cual busca generar un sistema avanzado para analizar datos extraídos desde redes sociales para obtener información relevante para las empresas en relación a sus productos y servicios. Específicamente, uno de los hitos que contempla aquel proyecto es el desarrollo de un clasificador de ironía en texto, a modo de disminuir el impacto de los falsos positivos provocados por la ironía en este texto y también constituir un elemento diferenciador frente a la competencia existente.

1.3 Objetivos

El siguiente trabajo busca alcanzar los siguientes objetivos:

1.3.1 Objetivo General

Diseñar e implementar un módulo clasificador de texto que, utilizando técnicas de Opinion Mining y Machine Learning, permita identificar la presencia de ironía en el contenido generado por usuarios de Twitter.

1.3.2 Objetivos Específicos

1. Estudiar el estado del arte actual respecto al análisis de ironía dentro del contexto de Opinion Mining, junto con otros elementos teóricos necesarios para el desarrollo del trabajo.
2. Determinar un método que permita dar una estructura definida a las opiniones extraídas para el estudio.
3. Definir una metodología basada en Opinion Mining para realizar el análisis de ironía y desarrollar el corpus lingüístico necesario.
4. En base al estudio previo, implementar un módulo que utilizando Machine Learning, permita la identificación de ironía en las opiniones analizadas.

[1] http://wi.dii.uchile.cl/

5. Desarrollar un proceso de experimentación para probar el desempeño del método implementado.
6. Evaluar los resultados obtenidos para generar conclusiones en torno al trabajo realizado y propuestas de trabajo futuro.

1.4 Hipótesis de Investigación

La detección automatizada de ironía en texto es un tema de investigación relativamente nuevo, y más aún considerando a la red social Twitter como fuente directa de datos. Sin embargo, en los últimos años se han propuesto metodologías interesantes y completas para cumplir este propósito. La hipótesis que motiva el desarrollo de este trabajo señala que es posible detectar la presencia de ironía en texto en idioma Español con cierto nivel de precisión, utilizando una adaptación de la metodología propuesta por Reyes et al. (2013) en [5] la cual involucra la construcción de un corpus haciendo provecho de la estructura de Twitter junto con la capacidad de las personas para detectar ironía, y además se basa en la utilización de un conjunto de diversos atributos que incluyen elementos sintácticos y semánticos del lenguaje y otros elementos asociados a la percepción psicológica de las personas al momento de expresarse, lo cual representa un factor diferenciador respecto a otras metodologías.

1.5 Metodología

Para lograr los objetivos planteados, se propone la siguiente metodología de trabajo:

Como primer paso se pretende estudiar el estado del arte y los elementos relacionados. En esta etapa se busca realizar una completa documentación respecto a Opinion Mining con el fin de familiarizarse con los elementos básicos que compondrán el trabajo a realizar. Además, interesa contar con conocimientos profundos respecto al estado del arte en relación a la identificación de ironía en texto.

Como segundo paso se busca determinar elementos característicos de la ironía presente en texto. En esta etapa se pretende complementar el estudio inicial con el objetivo de caracterizar la ironía usada en las opiniones, lo que mejorará el nivel de comprensión del problema y dará luces respecto a cómo podría identificarse.

Como tercer paso, se busca estudiar diversas metodologías utilizadas en el análisis de ironía para su posterior aplicación. En esta etapa se pretende identificar mediante el conocimiento del estado del arte de la identificación de ironía, aquellas metodologías que hayan dado resultados positivos. Como se verá en el marco conceptual, hoy en día no existen soluciones globales para la identificación de ironía dada la complejidad del problema, pero existen diversos intentos de distintos investigadores interesados en el tema. Estos intentos representan oportunidades para dar con mejores criterios de identificación.

Como cuarto paso, se llevará a cabo la implementación de un método que contenga las mejores prácticas analizadas. En esta etapa se pretende generar un método basado en las mejores prácticas ya probadas. Esta etapa también consiste en la implementación del clasi-

ficador, lo que involucra la aplicación de algoritmos y herramientas conocidas en relación al análisis de texto sacado de la Web. Este paso también involucra obtener resultados y evaluar desempeño de implementación. Es necesario realizar experimentos en base a la implementación hecha, para evaluar el desempeño de los métodos utilizados. Este paso permitirá determinar el grado de verosimilitud con el que es posible identificar ironía según el método definido previamente.

Finalmente, se desarrollarán conclusiones y determinar posible trabajo futuro enfocado a mejorar el proyecto. Se pretende concluir el trabajo en base al análisis de los resultados obtenidos previamente, y además definir las directrices necesarias para un eventual trabajo posterior respecto al tema.

1.6 Resultados Esperados y Alcances

Dentro de los resultados esperados de este trabajo, se encuentran los siguientes:

- La elaboración de un marco conceptual sólido que abarque las tendencias recientes respecto a la investigación de la detección automática de ironía en texto. Este campo de investigación es relativamente nuevo por lo que es fundamental para el proyecto y su trabajo futuro, contar con documentación sólida al respecto.
- La construcción de un corpus que englobe los criterios de clasificación determinados en el desarrollo del estudio y que permitan caracterizar la ironía con cierto nivel de precisión.
- Un módulo clasificador que permita identificar ironía en textos extraídos desde la red social Twitter, bajo cierto nivel de certeza y que permita obtener conclusiones acerca de las posibilidades existentes respecto al tratamiento de esta limitante. Como se ha mencionado anteriormente, la investigación asociada a esta problemática es bastante reciente por lo que dentro del marco de un proyecto que pretende obtener conocimiento relevante del contenido web, es importante saber si la identificación de ironía representa un área de estudio significativo para el futuro de la aplicación de Opinion Mining.

El siguiente proyecto abarcará el problema del análisis e identificación de ironía en texto, en base a las características de ironía capturadas por el modelo utilizado, las que no necesariamente se encuentran acotadas por definiciones tradicionales. Como se muestra más adelante, la ironía es un fenómeno complejo de definir.

Además, el proceso de análisis de texto será desarrollado utilizando técnicas ya conocidas y documentadas. Como se menciona en el Capítulo 2, los estudios actuales del problema apuntan al uso de criterios de identificación que sean compatibles con los métodos conocidos de análisis de texto ya que no existen métodos especializados o algoritmos específicos que ofrezcan soluciones directas a este problema.

Finalmente, el proyecto pretende dar con la implementación de un clasificador de texto preparado para el análisis de documentos de texto extraídos desde Twitter. La implementación utilizará técnicas de aprendizaje supervisado. Por limitaciones asociadas al tiempo, es

posible que el clasificador no se encuentre preparado para el análisis de contenido extraído de otros entornos, sin embargo se espera que los criterios de clasificación encontrados puedan ser adaptados para ser utilizados de forma general en un trabajo posterior.

1.7 Estructura del Informe

Este trabajo está compuesto por 7 capítulos distribuidos de la siguiente forma: El presente Capítulo 1 contiene una introducción al trabajo a realizar en torno a este proyecto de investigación aplicada.

El Capítulo 2 contiene un marco conceptual a modo de entregar al lector un acercamiento básico a los distintos conceptos teóricos que fundan las bases de este trabajo, así como un estado del arte respecto a las tendencias del último tiempo respecto al tema.

Luego, en el Capítulo 3 de describe el procedimiento y los criterios utilizados para la construcción del corpus lingüístico que se utilizará para la posterior clasificación de texto. Este corpus incluye los datos que se utilizarán para el entrenamiento de un algoritmo de aprendizaje supervisado. La construcción incluye recolección de datos, pre-procesamiento asociado a dar solución a problemas identificados en ellos y un posterior proceso de validación de los datos recolectados utilizando evaluadores humanos.

Posteriormente en el Capítulo 4 se describe en detalle el modelo a utilizar para caracterizar los atributos que se pretende identificar para realizar la detección de ironía, y además se describirá el algoritmo de clasificación que se utilizará para ello.

En el Capítulo 5 se describe todo el proceso de implementación desarrollado para poder extraer los atributos del modelo y para poder realizar la clasificación. Se realiza una revisión de las herramientas tecnológicas utilizados, se presenta el modelo de datos que da soporte a la extracción de atributos y finalmente se describe en forma detallada la implementación de la extracción de cada atributo, junto con las herramientas que utiliza.

El Capítulo 6 presenta los resultados obtenidos con la clasificación por medio de tablas y gráficos junto con un análisis exploratorio de los datos presentados. Los resultados incluyen la extracción de atributos y el proceso de pruebas del método de clasificación, el cual se realiza a través de dos pruebas distintas. Finalmente, se incluye un análisis crítico en función de los resultados obtenidos.

Finalmente el Capítulo 7 contiene las conclusiones finales generadas en base a la implementación realizada, los resultados obtenidos y las implicancias que se traducen de éstos. Por último, se propone el trabajo futuro que surge de estas conclusiones finales.

Capítulo 2

Marco Conceptual

El siguiente marco conceptual busca dar un acercamiento general al lector, con respecto a los tópicos que fundan las bases del desarrollo de este proyecto. Por lo tanto, se da una introducción al concepto de la Web y su relación con la generación de contenido por parte de las personas, el concepto de Data Mining y su aplicación sobre éste contenido, lo cual se enmarca dentro del concepto de Web Mining. En particular, para este trabajo interesa el estudio de Web Opinion Mining en relación al Análisis de Sentimientos y los efectos que la presencia de ironía tiene sobre éste, por lo que además se incluye un breve acercamiento al concepto de ironía en el lenguaje y un estado del arte respecto a la detección de ironía presente en el texto extraído desde el contenido generado por los usuarios de la Web.

2.1 La Web y el contenido generado por sus usuarios

La World Wide Web (WWW), propuesto por Tim Berners-Lee en 1990 [8] se define como un sistema de documentos de hipertexto (texto mostrado en una pantalla con referencias a otros documentos mediante hipervínculos, con los que el usuario puede interactuar) interrelacionados, accesibles mediante internet. En la actualidad enmarca uno de los principales usos que las personas le dan a internet y para el contexto de estudio en este trabajo representa la fuente principal de los datos a utilizar.

2.1.1 World Wide Web en sus inicios

Inicialmente, la Web 1.0 se utilizaba para compartir información de sólo lectura mediante páginas estáticas con actualizaciones periódicas y poco frecuentes. Las empresas utilizaban el medio para dar información a sus clientes (una especie de catálogo electrónico) y para tener presencia en internet. Además, el acceso a internet en un principio era limitado. Por ejemplo, en Estados Unidos hasta 1996 el costo del acceso residencial a internet era basado en la cantidad de tráfico y horas que el usuario consumía [9]. En Chile, el internet recién comenzaba a ser utilizado en forma masiva el año 2000, con un 16,6% de la población conectada.[1]

[1]Fuente: International Telecommunication Union - Percentage of Individuals using the Internet.

2.1.2 Web 2.0

El concepto de Web 2.0 fue definido oficialmente en 2004 por Dale Dougherty, vicepresidente de O'Reilly Media. Actualmente éste se utiliza para caracterizar sitios web que utilizan tecnología que supera la visión previa de sitios web estáticos donde los usuarios solo pueden acceder a los datos e información aportados por el creador de la página. Esta "nueva versión" sugiere un modo distinto de entender cómo se construyen los sitios web y cómo son utilizados por las personas.

Ahora los usuarios no interactúan con los sitios de forma unidireccional, si no que éstos ahora son capaces de leer contenido y además de escribir su propio contenido por lo que se considera que la interacción es bi-direccional. Dentro de sus características principales [10] se encuentran una experiencia de usuario más completa, el énfasis en el usuario y su participación, su escalabilidad y la generación de contenido dinámico entre otras. Una de las consecuencias directas de esto es que ahora la Web 2.0 da soporte a la colaboración y a la recolección de inteligencia colectiva al permitir que los usuarios mismos produzcan contenido, a diferencia de la Web 1.0 donde todo el contenido era estático, manejado por un administrador o *webmaster*. Dentro de los servicios generados gracias a estas características, es posible encontrar los siguientes [11, 12]:

- Blogs: Corresponden a páginas web donde un usuario genera *entradas* o posts basados en contenido propio a modo de revista o bitácora, presentados de forma cronológica desde el más reciente al más antiguo. Los visitantes de un blog pueden añadir comentarios a las entradas. En un principio los blogs se construían con contenido textual pero en la actualidad existen blogs basados en fotos, audio y video. Los posts de un blog puede categorizarse con el uso de *keywords* para facilitar la búsqueda y referencia de contenido.

- RSS (Rich Site Summary o Really Simple Syndication): RSS es un formato usado para compartir contenido de blogs o páginas web. Utiliza archivos XML que reúnen información y enlaces a sus fuentes. Mediante este formato, los usuarios son informados respecto a la actualización de los blogs o páginas web en las que tienen interés.

- Wikis: Las Wikis son páginas web que pueden ser editadas fácilmente por cualquiera que acceda a éste, promoviendo la generación de conocimiento colectivo. Además, para evitar actualizaciones malintencionadas del contenido, versiones históricas anteriores de éste pueden ser restaurados.

- Comunidades de contenido: Aparte de las Wikis donde los usuarios comparten información, existen otros sitios web donde sus usuarios aportan, organizan y comparten distintos tipos de contenido basado en fotos o videos.

- Foros/Bulletin Boards: Sitios donde los usuarios intercambian ideas e información, usualmente en torno a intereses comunes.

- Redes sociales: Corresponden a que permiten a sus usuarios construir perfiles personales, accesibles a otros usuarios con quienes intercambia contenido personal y comunicación.

2.1.3 Principios fundamentales de la Web 2.0

Los factores clave respecto a la innovación que las nueva aplicaciones basadas en la Web 2.0 han impuesto, pueden ser resumidos en tres principios fundamentales [12]:

1. Un enfoque en soluciones basadas en servicios, simples y *open source* en forma de aplicaciones web.
   Las aplicaciones web ahora son independientes de una plataforma y su desarrollo involucra software *open-source* de forma frecuente. Son ofrecidas como servicios en vez de como software empaquetado, fáciles de descargar, compartir y distribuir. Además son más simples, con interfaces de usuario menos saturadas ofreciendo un número de atributos más limitado pero dirigidos hacia los elementos que capturan el valor percibido por sus usuarios.
2. Nuevos modelos de negocios basados en servicios y nuevas oportunidades para acceder a clientes pequeños y productos en baja escala.
   Existen aplicaciones que forman parte de un modelo de negocios basado en suscripciones. Generalmente se ofrece un servicio básico de forma gratuita pero además existe un servicio *premium* donde se cobra a los usuarios una cierta cantidad para poder acceder a servicios más avanzados. Esto se complementa con servicios de publicidad y/o auspiciadores, donde se arrienda parte del espacio en un sitio web para mostrar publicidad a sus visitantes. Además se presentan nuevas oportunidades para acceder a nichos de clientes individuales, con intereses más específicos y personalizados. Sitios como eBay (www.ebay.com), Amazon (www.amazon.com) o MercadoLibre (www.mercadolibre.com) han aprovechado estos nichos mediante sus plataformas de *e-commerce*, abiertas a cualquiera.
3. Desarrollo de aplicaciones de forma continua e incremental, requiriendo la participación e interacción de los usuarios en formas nuevas.
   Los usuarios ahora pueden participar activamente del desarrollo de las aplicaciones Web 2.0, lo que ofrece diversos beneficios. Utilizando el feedback generado por los usuarios, se realiza un continuo proceso de mejoras y desarrollo en tiempo real de las aplicaciones. Nunca hay una versión definitiva (a lo que se le llama un "beta permanente") ya que el software se mantiene en desarrollo y perfeccionamiento en todo momento durante su existencia. Finalmente cada usuario puede añadir valor al incrementar la cantidad de inteligencia colectiva presente. Entre más usuarios participan, más avanzado y valioso es el servicio.

Este último punto es un elemento central de la Web 2.0 ya que los usuarios no solo actúan como consumidores sino que como generadores de contenido. Cada vez son más los servicios que aprovechan las funcionalidades de la Web 2.0. Desde foros de comunidades temáticas donde los usuarios discuten entre ellos respecto a temas de interés común, hasta sitios de *e-commerce* donde los usuarios comparten su experiencia e impresiones respecto a la compra y uso de diversos bienes de consumo. Estas características son de interés para el presente estudio ya que la constante generación de contenido dinámico gracias a la participación de los usuarios en la Web, es lo que nos permite recopilar datos para obtener conocimiento útil.

2.2 Knowledge Discovery y Data Mining

El crecimiento de que ha tenido Web en las últimas décadas respecto a su magnitud y a la variedad de su contenido, hoy en día representa una oportunidad para la obtención de conocimiento relacionado a diversos ámbitos. Este conocimiento puede obtenerse a través del proceso KDD, sigla para *Knowledge Discovery in Databases* que en Español significa *Descubrimiento de Conocimiento en Bases de Datos*, el cual corresponde al proceso de obtener patrones útiles y comprensibles a partir de grandes y complejos conjuntos de datos. Este proceso dispone del *Data Mining* ya que con sus herramientas es posible generar algoritmos que puedan desarrollar un modelo al explorar los datos, el que puede ser utilizado posteriormente para analizar otros datos y predecir elementos desconocidos a partir de éstos.

2.2.1 Knowledge Discovery in Databases

KDD es un proceso general que permite descubrir conocimiento útil a partir de un conjunto de datos. En [13] se define como *"El proceso no trivial de identificar patrones que sean validos, nuevos, potencialmente útiles y en última instancia comprensibles, en los datos"*. Esto implica obtener patrones, modelos, estructuras, etc. y que además sean válidos con cierto grado de certeza, nuevos para el sistema, útiles para las actividades realizadas y comprensibles, incluso luego de algún pre-procesamiento.

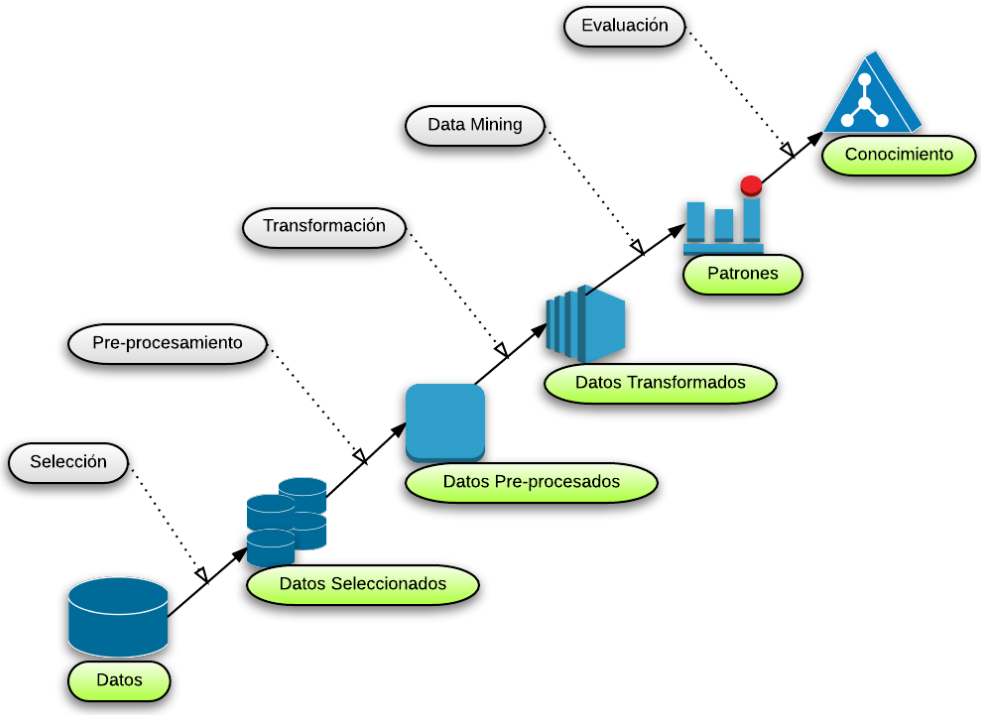

Figura 2.1: Proceso KDD
Fuente: Elaboración Propia.

Como se aprecia en la Figura 2.1, este proceso abarca el procedimiento completo de obtención de conocimiento relevante a partir de datos, desde cómo los datos son seleccionados

y almacenados hasta cómo sus resultados son interpretados y visualizados con énfasis en el descubrimiento de modelos o patrones desconocidos, pero con una visión más amplia que solo aplicar técnicas estadísticas para obtener estos resultados, apuntando a la automatización del análisis de los datos. El proceso puede ser caracterizado en nueve pasos, como se describe en [14]:

1. **Comprender el dominio de aplicación:** Es necesario analizar y entender aquello que se hará con el proceso, en función de los objetivos del usuario final y el entorno en el que se llevará a cabo el descubrimiento de conocimiento. A medida que el proceso avance, este análisis podría refinarse.
2. **Seleccionar y crear el set de datos objetivo del proceso:** Se deben determinar los datos que se utilizarán en el proceso. Esto implica ver su disponibilidad, obtener datos adicionales en caso de ser necesario y luego integrar todos éstos en un set que incluya todos los atributos que serán considerados en el proceso. Este paso es muy relevante ya que a partir del set resultante se construirán los modelos que se pretende descubrir y si algún atributo importante no está presente, se podría comprometer los resultados del estudio.
3. **Pre-procesamiento y limpieza:** Se mejora la fiabilidad del set de datos seleccionado. Esto incluye clarificar los datos, tratando valores vacíos o eliminando ruido y datos no significativos. Dependiendo de la naturaleza de los datos y del estudio, este paso podría involucrar métodos estadísticos complejos e incluso técnicas de predicción.
4. **Transformación de datos:** Se preparan y mejoran los datos para aplicar Data Mining sobre ellos. Esto puede abarcar reducción de dimensiones, transformación de atributos. Este paso es crucial, pero los métodos de transformación requeridos dependen del proyecto en específico.
5. **Selección de tareas de Data Mining:** Se debe decidir que tipo de Data Mining se utilizará, por ejemplo, clustering, regresiones o clasificación. Esto dependerá de las metas del proceso KDD y del desarrollo de los pasos anteriores. Las dos metas principales del Data Mining son la predicción, la que desarrolla en base a técnicas supervisadas y la descripción, alcanzada a partir de métodos no supervisados y de visualización. La mayor parte de las técnicas se basan en métodos supervisados de aprendizaje, donde un modelo es construido explícita o implícitamente generalizando sobre un número suficiente de muestras de entrenamiento, bajo el supuesto de que este modelo entrenado será aplicable a casos futuros.
6. **Selección de algoritmos de Data Mining:** Una vez decidido el tipo de Data Mining, se debe seleccionar las técnicas a utilizar. Esto implica escoger qué método específico se utilizará para la búsqueda de patrones, considerando las condiciones en las que éstas técnicas son más apropiadas.
7. **Aplicación de Data Mining:** Corresponde a la implementación de los algoritmos de Data Mining seleccionados. Será necesario emplear los algoritmos repetidas veces hasta obtener el resultado deseado, mientras se ajustan los parámetros de control de éste.
8. **Evaluación:** Se debe evaluar e interpretar los patrones descubiertos, en relación a las metas establecidas al inicio del proceso. Este paso se enfoca en la comprensión y utilidad del modelo inducido.

9. **Usando el conocimiento descubierto:** Finalmente se incorpora el conocimiento descubierto en el proceso. La idea es incorporarlo al entorno en que se pretendía aplicarlo y medir su impacto. Uno de los desafíos de esta etapa final es alejarse de las condiciones experimentales en las que el conocimiento fue generado, donde se tenía evidencia del conocimiento en un contexto estático (por ejemplo, en base a una muestra) para llevarlo a un contexto real de aplicación que podría estar sujeto a cambios.

2.2.2 Data Mining

Data Mining se puede definir como el proceso de descubrir patrones a partir de grandes cantidades de datos, cuyas fuentes incluyen bases de datos, *data warehouses*, repositorios de información, la Web o datos que son generados en forma dinámica. Es necesario señalar que si bien las técnicas de Data Mining permiten descubrir patrones presentes en un conjunto de datos, la utilidad que estos patrones representen se determinan en un paso posterior, a través del proceso KDD [15].

Existen distintas funcionalidades asociadas a los métodos de descubrimiento de Data Mining, cada uno con sus propios fines y su aplicación permite especificar el tipo de patrones que pueden encontrarse en el proceso. Es posible identificar métodos *descriptivos*, los cuales permiten caracterizar propiedades de los datos en un set determinado. Entre éstos se puede encontrar: *Caracterización* y *Discriminación* de datos, donde se realiza un resumen o abreviación (del inglés *summarization*) de los atributos generales de los datos y se realiza una comparación de estos atributos generales con los atributos generales de una o más clases de contraste definidas respectivamente. *Análisis de patrones frecuentes* mediante reglas de asociación y correlación. *Clustering* que permite identificar clases previamente desconocidas entre datos sin considerar etiquetas previas (las que generalmente no existen al aplicar este método), agrupando los datos según los principios de máxima similitud dentro de una clase y mínima similitud entre clases.

Así mismo, existen métodos *predictivos* que a partir de un set de datos o muestra, buscan construir automáticamente un modelo de comportamiento y predecir el valor de una o más variables relacionadas con la muestra. Entre éstos están: *Clasificación* el cual es el proceso de encontrar un modelo o función a partir del análisis de un set de datos de entrenamiento con clases conocidas (datos previamente etiquetados), que describa y distinga entre clases de datos. Luego, el modelo es usado para predecir la clase o etiqueta de datos en los que no se conoce. *Regresión* se utiliza para predecir valores numéricos desconocidos o no existentes en base a un método estadístico y es utilizado frecuentemente en la predicción numérica o también para identificar tendencias de distribución basado en los datos disponibles.

Otra funcionalidad relevante corresponde a la identificación de *outliers* o parte de un set de datos que no es consistente con las características del resto del set. Éstos pueden detectarse utilizando pruebas estadísticas que asuman cierta distribución de probabilidad para los datos o usando medidas de distancia en la que los objetos lejanos al resto de los datos son considerados outliers.

2.3 Web Mining

La aplicación de las técnicas de Data Mining descritas anteriormente para descubrir nuevos patrones en la Web se denomina *Web Mining*, y gracias al explosivo crecimiento de la Web y sus fuentes de datos, este campo ha sido objeto de interés considerable debido a todas las posibilidades que provee el descubrimiento de conocimiento y además su evolución el área involucra una gran cantidad de desafíos.

2.3.1 Definición

Web Intelligence se define como el uso de técnicas avanzadas de Inteligencia Artificial y Tecnologías de Información para el propósito de explorar, analizar y extraer conocimiento del contenido web [3]. Esto se realiza con el objetivo de crear una nueva generación de productos y servicios basados en el internet. Estas técnicas avanzadas se asocian al Web Mining. Formalmente Cooley et al. (1997) en [16] define Web Mining como el descubrimiento y análisis de información útil desde la World Wide Web. En su taxonomía es posible encontrar diversos enfoques, apuntando al estudio de las diversos tipos de datos relacionados a la Web y al uso por parte de los usuarios. En [17] se caracterizan tres categorías distintas basado en el tipo de dato que es posible extraer en el proceso: *Web Usage Mining*, *Web Structure Mining* y *Web Content Mining*. A continuación se describen algunas de las características generales de cada categoría y se nombran algunas de sus aplicaciones [17, 18].

2.3.2 Categorías

- **Web Usage Mining**
  Se enfoca en analizar aquello que los usuarios buscan ver en internet o cualquier otro registro de su actividad, estudiando el comportamiento de éstos durante el uso de los sitios web con el objetivo de generar perfiles de usuario. Estos datos pueden ser obtenidos desde servidores web, proxies, aplicaciones del cliente (como navegadores web y sus historiales o cookies) y de ellos puede obtenerse conocimiento asociado al uso que los usuarios hacen de cierto sistema o de los intereses de éstos. Este conocimiento puede aplicarse en personalización de sistemas web, marketing, diseño y evaluación de sitios web e incluso como apoyo para tomar decisiones al respecto.
- **Web Structure Mining**
  Estudia la estructura los sitios web, buscando potenciales modelos subyacentes en éstas. Esto es extrayendo patrones a partir de los enlaces existentes entre sitios (entrantes o salientes) o minando la estructura de los documentos (estructuras HTML o XML). Esto puede usarse para inferir información acerca de los sitios, como su relevancia en función de referencias o citas entre ellos mediante el uso de algoritmos que determinan un puntaje en base a estos parámetros. Otra aplicación posible es analizar la Web como un grafo construyendo las distintas conexiones entre sitios, evidenciando concentraciones de ellos y sus características.
- **Web Content Mining**
  Se refiere al descubrimiento de información relevante de los contenidos de la Web, los cuales incluyen texto, imágenes, audio y video. La mayor parte de los documentos de la

Web son de texto y en éstos se aplican distintas técnicas como *Text Mining*, clustering y clasificación de texto, entre otros. Sus resultados tienen diversas aplicaciones como clasificación de documentos web, desarrollo de motores de búsqueda, sistemas de recomendación, sitios web adaptativos, etc. En particular, Text Mining es un sub-campo del Data Mining relacionado con la extracción de conocimiento a partir de documentos de texto y su aplicación en el contexto de Web Content Mining considerando el gran volumen existente de documentos de texto en la Web, hace que ésta sea una importante fuente de recursos para estudios lingüisticos.

2.3.3 Algunas aplicaciones

La aplicación de Web Mining tiene múltiples usos en conjunto con otras herramientas de áreas como Procesamiento de Lenguaje Natural y Machine Learning. Un ejemplo en el que se aplica Web Content Mining corresponde al trabajo de Marrese en [19], quien pretende utilizar específicamente Opinion Mining (concepto que surge de Web Content Mining como se verá posteriormente) en el análisis de reviews hechas por turistas en la X región de Chile para predecir sus preferencias sobre productos turísticos. Otro ejemplo corresponde al trabajo realizado por Tapia en [20] donde pretende implementar un clasificador de textos extraídos desde Twitter, según su polaridad (Positiva o Negativa), utilizando Opinion Mining y Machine Learning. Cabe mencionar que estas aplicaciones no se limitan solo al análisis de texto. Por ejemplo, Martínez en [21] utilizando herramientas relacionadas con Web Mining, desarrolla un método de identificación de *website keyobjects* en función del análisis de señales fisiológicas, específicamente la dilatación pupilar, medido por un equipo de *eye-tracking*. Los ejemplos citados anteriormente fueron desarrollados en torno a proyectos creados al interior del WIC[2], en la Universidad de Chile.

2.4 Web Opinion Mining

El interés en identificar ironía presente en texto tiene su origen en el siguiente concepto. En el contexto de Web Content Mining es posible enfrentarse a diversos tipos de contenido. Entre ellos, existen las opiniones o impresiones que cada día los usuarios registran en los distintos medios web respecto a variados temas. A continuación se dará un breve acercamiento al concepto de Opinion Mining, sus aplicaciones y los desafíos que se identifican en la actualidad en torno a este campo de investigación.

2.4.1 Opinion Mining o Sentiment Analysis y su potencial

Opinion Mining o *Sentiment Analysis* (Análisis de Sentimientos) corresponde al estudio computacional de opiniones, sentimientos, subjetividad, evaluaciones, actitudes, emociones, etc. expresados en texto [6]. Ambos términos tienen diferencias sutiles, pero generalmente se utilizan para describir el mismo campo de estudio. Es importante recalcar que los términos *opinión* y *sentimiento* no son equivalentes, pero a pesar de esto tanto Opinion Mining como Análisis de Sentimientos se enfocan principalmente en opiniones que expresan sentimientos positivos o negativos.

[2]http://wi.dii.uchile.cl/

La importancia de las opiniones radica en el impacto que éstas tienen nuestro comportamiento. Las creencias y percepciones de nuestra realidad están fuertemente condicionadas a como otros ven esta realidad. Un ejemplo de esto es la toma de decisiones. Cuando una persona toma una decisión, usualmente buscan opiniones en los demás, lo que aplica tanto a individuos como a las grandes organizaciones. En el pasado, existían límites asociados a los círculos de personas cercanas para poder recurrir a otras opiniones y en el caso de las empresas, solo tenían acceso a métodos como encuestas, focus groups, consultoría externa, etc. Pero, con el crecimiento de los medios presentes en la Web como blogs, foros de opinión y redes sociales, estos límites han ido desapareciendo. Gracias a las características de la Web 2.0, las personas pueden compartir sus opiniones respecto a diversos temas con otros usuarios. Hoy en día es posible encontrar este tipo de contenido en grandes cantidades y las personas ya no están limitadas a las opiniones de sus círculos para poder apoyar su toma de decisiones.

Esta demanda de información va en crecimiento a medida que el contenido presente en la Web también crece, y sus fines abarcan desde la elección y compra de productos o servicios hasta información política o sobre temas de contingencia social. El interés de los usuarios en las opiniones de los demás y su potencial influencia sobre sus decisiones, es algo que las empresas y organizaciones están tomando cada vez más en cuenta. Por lo tanto, aparte de las personas mismas existe un nicho interesado en disponer de sistemas capaces de analizar de forma automática las opiniones generadas por clientes para poder obtener conocimiento a partir de éstas [4].

2.4.2 Aplicaciones

Como se ha mencionado, el contenido web es variado y crece constantemente por lo que las aplicaciones de Opinion Mining también son variadas. Es posible identificar las siguientes:

- **Sitios web de Reviews**
  Un *review* corresponde a un estudio general respecto a algo, a modo de crítica o evaluación. Los sitios de reviews abundan en la Web, por lo que resulta interesante pensar en métodos que permitan obtener un resumen de las opiniones generadas en estos reviews de forma automática para generar conocimiento accesible a los demás o incluso corregir errores cometidos por usuarios al momento de entregar una evaluación (por ejemplo si un usuario entrega un review positivo respecto a algo pero accidentalmente selecciona una calificación baja).
- **Uso complementario**
  El Análisis de Sentimientos podría ser utilizado como un complemento habilitador para otras tecnologías. Por ejemplo, se podrían mejorar los sistemas de recomendación, filtrando los elementos a recomendar para los distintos usuarios en función del feedback que cada elemento haya acumulado. También se podría utilizar para detectar lenguaje no deseado o mensajes hostiles en los distintos medios de comunicación de la Web. Respecto a los anuncios que se publican en los sitios web, se podría detectar el tipo de contenido que cada página para seleccionar anuncios adecuados y evitar mostrar publicidad con contenido inapropiado para ciertos sitios.

- **Business Intelligence**
  La posibilidad de poder aplicar Opinion Mining para construir herramientas dedicadas al Business Intelligence es uno de los principales factores de interés por parte de empresas y organizaciones en este campo de investigación. Las tecnologías de Análisis de Sentimientos utilizadas para extraer opiniones de documentos no estructurados y generados por humanos podría representar una potente herramienta para poder responder a preguntas complejas relacionadas con el negocio y la percepción que tienen los clientes respecto a los productos y servicios ofrecidos. Analizar directamente a los clientes podría ser costoso y difícil, por lo que sería mejor contar con un sistema que obtenga de las distintas fuentes disponibles las reviews u otras opiniones de los clientes para luego generar una evaluación o indicador que agregue toda la información obtenida de ellas, ahorrando tiempo y costos.
- **Otros dominios**
  Podrían surgir otras aplicaciones útiles en distintos dominios. Por ejemplo, aplicarlo en análisis político para obtener una idea de lo que piensan los votantes en una elección, el apoyo o rechazo a los candidatos y también para mejorar la calidad de la información disponible para la gente. Otro alcance se relaciona con el concepto de *e-rulemaking*, analizando automáticamente las opiniones de la gente respecto a las propuestas de regulación gubernamental o materia de leyes. Por último, es posible señalar aplicaciones sociológicas, por ejemplo estudiando cómo se difunden las ideas y la innovación y su relación con la disposición positiva o negativa hacia éstas, indicando quienes serían más o menos receptivos a esta información.

2.4.3 Enfoques de Análisis

En función de las distintas aplicaciones identificadas para Opinion Mining, existen variados enfoques de análisis basados en la clasificación de sus características. A continuación se describen brevemente los principales enfoques considerados [4]:

- **Análisis de Polaridad**
  El Análisis de Polaridad (también llamado Clasificación de Polaridad de Sentimientos) generalmente corresponde a una clasificación binaria de un texto que contenga alguna opinión, según una de las dos polaridades de sentimientos (positivo o negativo). Esta detección también puede incluir otras clases aparte de positivo o negativo, como opiniones neutra las cuales son interpretadas según la caracterización deseada. Algunos las consideran como información objetiva, otros como una mezcla de opiniones positivas y negativas, etc.
- **Detección de Subjetividad**
  También denominado Identificación de Opiniones, este enfoque se enmarca en el problema que surge al querer saber cuando un documento contiene una opinión (texto con información subjetiva) o no. Frecuentemente el Análisis de Polaridad asume que los documentos analizados corresponden a opiniones, pero en algunas aplicaciones es necesario decidir cuando cierto documento es una opinión o identificar las porciones de un documento que son subjetivas.

- **Análisis de Sentimientos y Tópicos**
  Un supuesto recurrente en el Análisis de Sentimientos corresponde a que cada documento analizado gira en torno a un tópico en particular, asumiendo que el set de análisis fue construido inicialmente en base a este tópico. Sin embargo, pueden existir interacciones entre los tópicos y sus opiniones que requieran considerar ambos análisis simultáneamente. Por ejemplo, pueden tenerse documentos que contenga opiniones relevantes pero que incluyan elementos que no estén relacionados con el tópico analizado, por lo que no deberían considerarse en ese análisis. Otro caso puede surgir cuando se tenga un documento que contenga información respecto a múltiples tópicos de interés para el usuario, por lo que sería útil separar las opiniones incluidas en él y asociarlas a cada tópico relevante (lo que podría pasar con análisis comparativos de productos relacionados o documentos que incluyan varios atributos de un producto).
- **Puntos de vista y Perspectivas**
  Al aplicar Análisis de Sentimientos en textos relacionados con política, el análisis no solo puede reducirse a la polaridad de las opiniones. Puede hacerse énfasis en actitudes en general, expresadas a través del texto y que no necesariamente apuntan a un tema en particular. Este tipo de trabajo trata de clasificar el texto en escalas relacionadas con los puntos de vista o las perspectivas expresadas a través del texto (por ejemplo, usando una escala de ideologías políticas para clasificar).
- **Otros Alcances**
  Existen otros alcances para el Análisis de Sentimientos, como la detección y reconocimiento de humor en el texto, la categorización por géneros lingüísticos, la clasificación según el estilo del autor de un texto, entre otros.

2.4.4 El problema de la Ironía

A pesar de sus múltiples aplicaciones y enfoques asociados, Opinion Mining es un campo de investigación reciente con una gran cantidad de desafíos y problemas que quedan por superar. De éstos, es de especial interés para el estudio la presencia de ironía en el texto generado por los distintos usuarios que aportan su opinión a través de la Web. Seerat y Azam en [7] señalan entre los distintos desafíos identificados, que las expresiones irónicas o sarcásticas llevan a opiniones engañosas. La existencia de estas expresiones pueden cambiar el sentido de una opinión, por lo que un texto con palabras positivas construido en base a expresiones irónicas en realidad tiene un sentido negativo de forma subyacente, por lo que si es clasificado como una opinión positiva debido a las palabras que la componen, la clasificación será errónea.

En particular, el análisis en redes sociales es susceptible a este problema. Maynard et al. (2012) en [22] señala que las redes sociales contienen un uso extensivo de ironía y sarcasmo, el cual es complejo de detectar de forma automática. También Liu en [6] menciona que las oraciones sarcásticas son difíciles de tratar debido a que es utilizada por usuarios que desean transmitir un mensaje contrario a lo que escriben literalmente, por lo que en el contexto de Análisis de Sentimientos tiene efectos negativos en la clasificación. Por este motivo, la detección automática de ironía ha empezado a ganar la atención de este campo de investigación, sobretodo cuando el objeto del análisis corresponde al contenido generado

por usuarios de la Web, o las opiniones transmitidas a través de redes sociales donde el uso de este recurso es frecuente. Filatova en [23] indica que las aplicaciones de Opinion Mining y Análisis de Sentimientos pueden mejorar su desempeño si se identifican correctamente las expresiones sarcásticas presentes en el contenido analizado.

Si bien este problema es mencionado por estos autores y por otros en torno a investigaciones relacionadas con aplicaciones de Opinion Mining, actualmente no existen estudios estadísticos conocidos que muestren cifras claras respecto al uso real de este tipo de expresiones en redes sociales ni del impacto que tendrá su detección en el desempeño de la aplicación de Opinion Mining, en parte debido al interés reciente respecto a este fenómeno y al hecho de que identificar ironía es complejo incluso para un humano. Sin embargo, basta con realizar un análisis exploratorio en redes sociales como Twitter para encontrar con frecuencia el uso de ironía a distintos niveles, ya sea con el fin de hacer bromas, generar críticas, comunicarse entre pares, etc. Esto tiene sentido cuando se analiza el rol que tiene este fenómeno en la comunicación, el cual será descrito posteriormente en este capítulo.

2.5 Machine Learning

Una vez que se ha establecido la importancia de detectar la presencia de ironía en el texto extraído desde la Web, es necesario disponer de las herramientas necesarias para ello. Detectar ironía de forma automática es un problema complejo sin una solución predeterminada. Si incluso para las personas percibir un mensaje irónico puede ser difícil, para una máquina la tarea debe ser aún más complicada. Sin embargo, existen distintas herramientas asociadas al Data Mining que pueden ser útiles. En esta sección se describirá a grandes rasgos el concepto de *Machine Learning* y su relación con el Text Mining y los algoritmos de aprendizaje supervisado, los cuales serán necesarios para el estudio.

2.5.1 Definición

Existen situaciones en las que nos gustaría disponer de alguna herramienta que nos permita obtener distintos resultados en función de datos de entrada o *inputs*. Generalmente esto se hace mediante un algoritmo, el cual mediante una secuencia de procesos es capaz de transformar aquel input en un resultado de salida o *output*. Existen diversos algoritmos para distintos fines como ordenar o buscar datos. Sin embargo, no existen algoritmos para todos los resultados que se desean obtener. Por ejemplo, no existen algoritmos que permitan predecir las compras futuras de un cliente en base a su historial de compras. Tampoco existen algoritmos que permitan conocer enfermedades futuras en un paciente, en base a su historial médico y las características de su estilo de vida. Pero, lo que si hay en abundancia son datos. Para muchas situaciones como estas se dispone de datos que se han ido registrando a través del tiempo y asociados a estos datos existe un proceso que los explica. En cada situación, algo está generando estos datos y este proceso no es al azar. Es posible reconocer patrones presentes en los datos y a su vez, llegar a una buena aproximación de los resultados que podría entregar el proceso en cuestión [24].

Entonces nos gustaría que mediante el análisis de estos patrones presentes en datos históricos, una máquina sea capaz de aprender a predecir el resultado de una situación, en

base a patrones que caractericen aquella situación. Esto corresponde al concepto de Machine Learning, el cual se puede definir como el proceso de programar una máquina o computador para que sea capaz de entregar resultados lo suficientemente útiles en base al uso de datos de ejemplo o experiencia pasada [24]. Estos resultados son posibles gracias al "aprendizaje" que haga la máquina en función de un modelo que caracterice los patrones que pretendemos identificar. En este estudio, el interés apunta a la aplicación de *aprendizaje supervisado* para realizar la clasificación de documentos de texto en base a los distintos atributos representativos presentes en ellos, por lo que se hará una revisión breve a los conceptos necesarios para su realización.

2.5.2 Aprendizaje Supervisado

El aprendizaje supervisado es uno de los enfoques de Machine Learning, en el que para transformar un input en cierto output con resultados relevantes, los valores para determinar ese output son proveídos por un *supervisor*, el cual es basado en un modelo o una función dependiente de ciertos parámetros y de un set de datos de entrenamiento, el cual ha sido previamente etiquetado con los valores correctos que se deberían obtener dado un input. Entonces durante el aprendizaje supervisado, los parámetros que forman parte del supervisor son optimizados dado los datos de entrada o input, para reducir al máximo el error del ajuste y que la predicción del resultado se acerque lo más posible a los valores correctos indicados por el set de entrenamiento utilizado.

Para la construcción del set de entrenamiento, se deben incluir datos con etiquetas indicando las distintas clases de datos que la máquina necesita *aprender* a identificar. En el caso más simple cuando se pretende realizar una clasificación de datos, se cuenta con dos clases de datos. Una clase se denomina *set positivo* y corresponde a un conjunto de datos que representan aquello que se pretende identificar o clasificar. La otra clase corresponde al *set negativo* el cual incluye ejemplos contrarios a lo que se pretende identificar [24]. Una vez que se cuenta con estos sets de entrenamiento, se debe identificar los atributos que caracterizarán las distintas clases para poder realizar el *entrenamiento* del algoritmo de clasificación, lo que le permitirá posteriormente diferenciar estas clases para poder realizar predicciones sobre nuevos datos de input. Este proceso de aprendizaje supervisado puede apreciarse en la Figura 2.2. La construcción y extracción de atributos requieren procesar el texto utilizando distintas herramientas asociadas a Data Mining para poder obtener la información relevante de ellos.

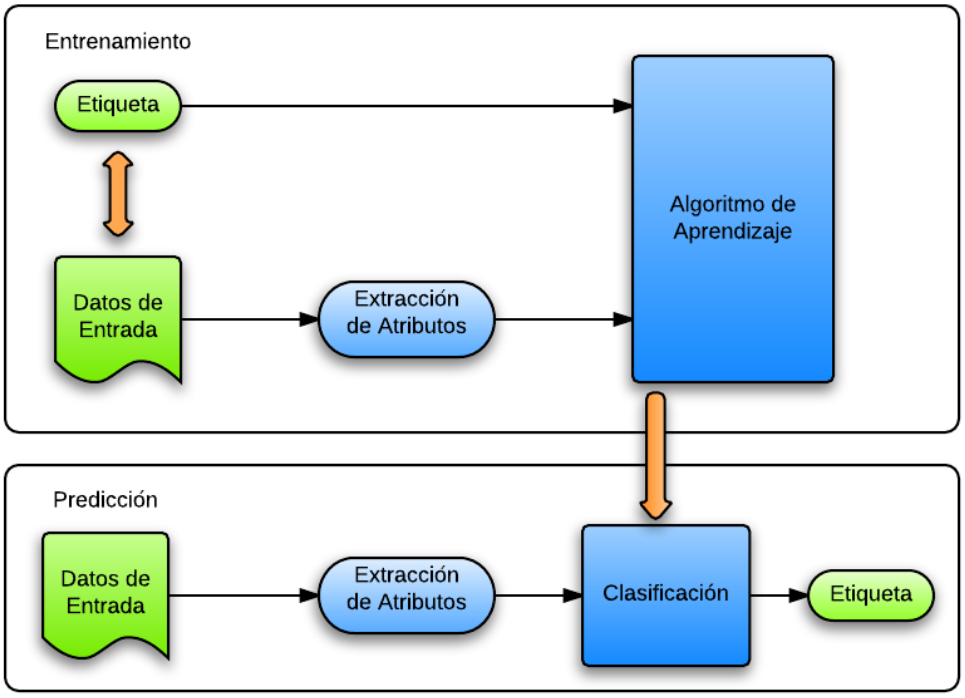

Figura 2.2: Proceso de Aprendizaje Supervisado
Fuente: Elaboración Propia.

2.5.3 Text Mining

Text Mining corresponde al descubrimiento de conocimiento a partir de documentos de texto, utilizando técnicas de *Information Retrieval* y de *Procesamiento de Lenguaje Natural* o NLP (del inglés Natural Language Processing) las cuales son vinculadas con el método de KDD y los algoritmos de Data Mining. Hotho et al. (2005) en [25] define Text Mining en función de tres áreas específicas relacionadas con su uso:

- **Extracción de Información**
  Este enfoque asume que Text Mining corresponde esencialmente a la extracción de información desde texto. Por ejemplo, extracción de hechos (*facts*) de un texto.
- **Proceso KDD**
  Otro enfoque describe el Text Mining como un proceso con una serie de pasos involucrando extracción de información, Data Mining y procedimientos estadísticos, entre otros. O sea, se ve como la aplicación de KDD en texto.
- **Data Mining de texto**
  Finalmente, se define Text Mining como la aplicación de algoritmos y métodos de Data Mining en documentos de texto, con el objetivo de encontrar patrones útiles. Para cumplir este objetivo, es necesario aplicar un pre-procesamiento a los textos analizados y para esto, muchos autores utilizan métodos de NLP u otros más simples para poder extraer datos desde los textos.

Procesamiento de Lenguaje Natural (NLP) corresponde a un área que investiga cómo utilizar un computador (máquina) para poder entender y manipular texto de lenguaje natural para realizar cosas útiles con éste [26]. El texto de lenguaje natural puede ser oral o escrito y puede estar escrito en cualquier lenguaje , género, etc. Su particularidad es que son textos generados por humanos, en un lenguaje utilizado por ellos para comunicarse y además éstos son extraídos directamente desde su uso y no con una estructura preparada para su análisis [27]. Como frecuentemente se trabaja con texto sin una estructura definida, es necesario tratar estos datos para poder utilizarlos.

2.5.4 Pre-procesamiento de texto

Los métodos de pre-procesamiento de texto son necesarios para obtener información a partir de conjuntos grandes de documentos. La mayor parte de las técnicas de Text Mining, consideran que un documento de texto puede ser expresado a través del set de palabras que contiene. Esta representación se denomina *bag-of-words*, y puede ser obtenida mediante este pre-procesamiento. A continuación, se describen los pasos asociados a éste [25, 28]:

- **Tokenización**
  Como paso inicial, se busca obtener todas los elementos que permitan caracterizar un documento de texto. Este proceso se denomina *Tokenización*, en el cual este documento es separado en las distintas unidades lingüísticas identificables en él. Estas unidades son llamadas *tokens* y corresponden a cadena de caracteres, los cuales se identifican según un criterio en particular. En general, el criterio utilizado corresponde a seleccionar palabras y eliminar otros elementos como signos de puntuación o cualquier otro caracter que no corresponda a una palabra. Sin embargo, según se requiera este criterio puede ser modificado. Los tokens pueden contener otros elementos dependiendo de las preferencias del análisis. Por ejemplo, emoticonos, abreviaciones, elementos numéricos (precios y porcentajes), entre otros [29]. El resultado de este proceso corresponde a un *diccionario* (arreglo de tokens) que contiene por separado los distintos tokens identificados.
- **Filtrado (Borrado de Stopwords)**
  El proceso de filtrado corresponde a la eliminación de elementos no deseados del diccionario resultante en el paso anterior. Un método estándar y ampliamente utilizado corresponde al borrado de *stopwords*. Las stopwords corresponden a palabras presentes con un alta frecuencia en los documentos de texto, pero que no agregan mayor información al contenido de un documento de texto más allá de su función gramatical, tales como pronombres, artículos, conjunciones, preposiciones, etc. Dadas sus características, estas palabras son eliminadas ya que no agregan valor a la información entregada por el documento y a la vez, su alta frecuencia puede representar ruido para los algoritmos basados en la frecuencia de palabras. En general, la estrategia para eliminar stopwords es utilizar una lista de éstos (*stoplist*) para luego comparar los elementos del diccionario con esta lista, eliminando las coincidencias entre ambos.
- **Lematización**
  Los métodos de Lematización apuntan a la transformación de palabras desde su flexión (alteración asociada al sentido gramatical de una palabra) a su forma base. Por ejemplo,

verbos conjugados son llevados a su infinitivo, o sustantivos son llevados a su forma singular. Para realizar este proceso debe realizarse un adecuado análisis morfológico al vocabulario utilizado, con el objetivo de conocer el rol de cada palabra en una oración o documento, lo que en general se realiza asignando *part-of-speech* a cada palabra. Este proceso demanda bastante tiempo y aún así suele contener errores, por lo que en la práctica se suele optar por métodos de *stemming*.

- ***Stemming***
  El proceso de Stemming también busca llevar las palabras a su forma base. Se basa principalmente en la construcción de *stems* o raíces de las palabras, los cual se lleva a cabo eliminando sus afijos (elementos adicionales adjuntos a un stem para cambiar su sentido gramatical) mediante un algoritmo. Esto convierte a todas las palabras que tengan una misma raíz a una forma base común.

2.5.5 Extracción de Atributos

Para poder obtener un resultado en función de los datos presentes mediante la aplicación de Machine Learning, es necesario caracterizar patrones que permitan a la máquina reconocer y capturar aquellos resultados útiles. Esta caracterización se realiza a través de la construcción de atributos a partir de los datos analizados. Luego de aplicar el pre-procesamiento necesario a los datos, se deben identificar aquellos atributos relevantes para la obtención de los resultados deseados. Estos atributos pueden ser diversos y dependerán de los objetivos finales detrás de esta captura de información. Pueden ir desde la captura de sufijos en un conjunto de nombres para la identificación de su género sexual hasta la frecuencia con la que se utilizan ciertas palabras para diferenciar opiniones de noticias, entre otros. Sin embargo, en referencia al análisis de sentimientos existen estructuras que son utilizadas con frecuencia para capturar ciertos patrones [30].

- **N-gramas**
  Corresponden a cadenas de largo N, en los que su unidad corresponde a cada palabra. Así, un texto puede ser dividido en estas cadenas para caracterizar la construcción de las frases que lo conforman (su contexto local), lo que a su vez permite construir una *historia* que eventualmente podría predecir características de este texto. En general se utilizan cadenas de dos a tres palabras (bigramas o trigramas) ya que entre mayor es el orden, la cantidad de parámetros aumenta considerablemente. En particular, estas estructuras son utilizadas en una gran cantidad de estudios relacionados con la categorización o clasificación de texto.
- **Part-of-Speech Tagging**
  Part-of-Speech Tagging (POS Tagging) corresponde a asignar a cada palabra dentro de un texto, una etiqueta morfosintáctica que represente su rol gramatical dentro de éste. Así es posible extraer información al reconocer patrones en el uso de sustantivos, verbos, adjetivos y adverbios en los textos objetivos, analizando gramaticalmente su estructura (parsing). En la Figura 2.3 se muestra un ejemplo de POS Tagging en Español, realizado con la herramienta *TreeTagger*[3] desarrollada en el Institute for Computational

[3]http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/

Linguistics de la Universidad de Stuttgart, el cual es utilizado en este estudio como se verá en el Capítulo 5.

El hombre está escribiendo su trabajo					
El	hombre	está	escribiendo	su	trabajo
ART	NC	VEfin	VLger	PPO	NC

Figura 2.3: Ejemplo de POS-Tagging de una oración
Fuente: Elaboración Propia.

- **Collocations**
  Corresponden a secuencias de dos o más palabras, cuyo significado no puede ser extraído directamente desde sus componentes sino que depende directamente de la combinación de palabras, por lo que pueden ser utilizadas para identificar elementos subjetivos en el texto. No es trivial reemplazar alguna de sus componentes, y debido a esto suelen ser dependientes del idioma en el que son expresados ya que es posible que la traducción literal de una combinación desde un idioma a otro no conserve el significado inherente a la combinación en el idioma original.

2.5.6 Métodos de Aprendizaje Supervisado

Los métodos de aprendizaje supervisado utilizan algoritmos que se valen de una serie de atributos presentes en Opinion Lexicons los cuales corresponden a listas de palabras con orientaciones de polaridades previamente etiquetadas. Estos pueden construirse en base a diccionarios o en base a corpus lingüísticos. Para tareas de clasificación, el aprendizaje supervisado utiliza estos datos previamente etiquetados en relación a lo que se busca clasificar, los que se dividen en un set de entrenamiento, donde se enseña al modelo a clasificar en base a los atributos y los datos conocidos, y un set de validación o prueba con el que se calibran los parámetros del modelo y se prepara para el trabajo sobre datos "crudos". A continuación se describen en forma breve algunos de los algoritmos de aprendizaje supervisado utilizados con frecuencia para tareas asociadas a la categorización de documentos de texto:

- **Naive Bayes**
  El algoritmo Naive Bayes corresponde a un clasificador estadístico, el cual predice la probabilidad de que un elemento pertenezca a cierta clase basándose en el Teorema de Bayes [15].

$$P(c_i \mid X) = \frac{P(X \mid c_i)P(c_i)}{P(X)} \tag{2.1}$$

  La ecuación 2.1 refleja los fundamentos del algoritmo basados en Bayes, donde dado un vector de atributos X, se obtiene la probabilidad de que pertenezca a una clase c_i. Para reducir el costo computacional del cálculo de la probabilidad *a posteriori* de X condicionado a c_i ($P(X \mid c_i)$), se asume como supuesto que esta probabilidad es independiente entre clases. De este supuesto (considerado algo ingenuo) surge el nombre del algoritmo y a pesar de esto, es ámpliamente utilizado debido a la simplicidad de su implementación y a los buenos resultados que obtiene.

- **Árboles de Decisión**
  Los árboles de decisión son diagramas de flujos en forma de árbol, el cual realiza una predicción de la clase a la que pertenece un vector de atributos dado X utilizando sus elementos. El árbol inicia con un *nodo raíz*, para dividirse de forma progresiva en distintos *nodos de decisión* por medio de *ramas*. Cada rama representa los datos resultantes del nodo de decisión (esto generalmente divide el conjunto de datos para discriminar entre las distintas clases) y a medida que los datos recorren el árbol, van trazando un camino a lo largo de éste. Eventualmente, los caminos posibles a lo largo del árbol desembocarán en un *nodo hoja*, el cual representa la clase a la que pertenecen los datos que llegan ahí [15]. En la Figura 2.4 se aprecia un ejemplo de árbol de decisión, donde los rectángulos representan los nodos de decisión, las flechas son las ramas y los nodos ovalados representan las hojas, donde se especifica la clase a la que pertenecen los datos evaluados.

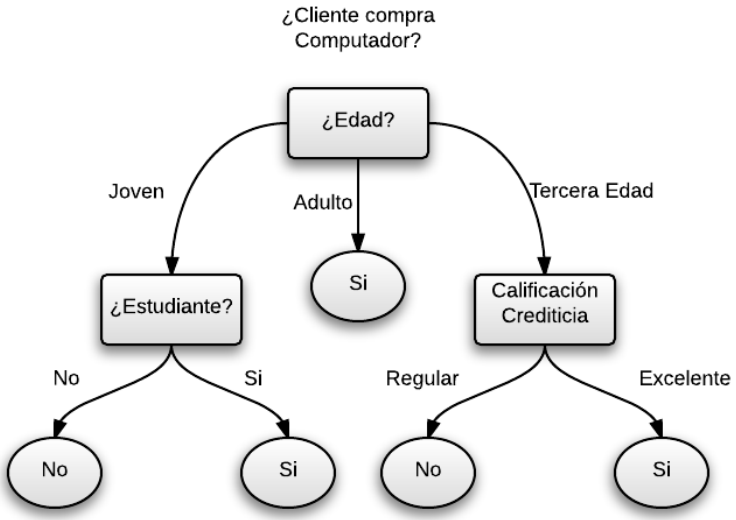

Figura 2.4: Ejemplo de un Árbol de Decisión para la predicción de clientes compradores de computador
Fuente: Ejemplo adaptado de [15].

- **Support Vector Machines**
  Corresponde a un método en el cual los datos de entrada son llevados a un espacio dimensional mayor, con el fin de que sean más separables (comparado con el espacio original). Para separarlos, se construye un *hiperplano* que divide los datos. Generalmente existen infinitos hiperplanos posibles para realizar esta división, por lo que se busca un hiperplano que mantenga una distancia de separación máxima, y para obtenerlo se utilizan otros dos hiperplanos paralelos a este de tal forma que al optimizar la distancia de separación de estos hiperplanos con el utilizado para la separación, se obtendrá un hiperplano con el máximo margen entre los datos y por lo tanto, la distancia entre éste y el dato más cercano será máxima [31]. Este hiperplano se denomina *Hiperplano de margen máximo* (en inglés, maximum marginal hyperplane).

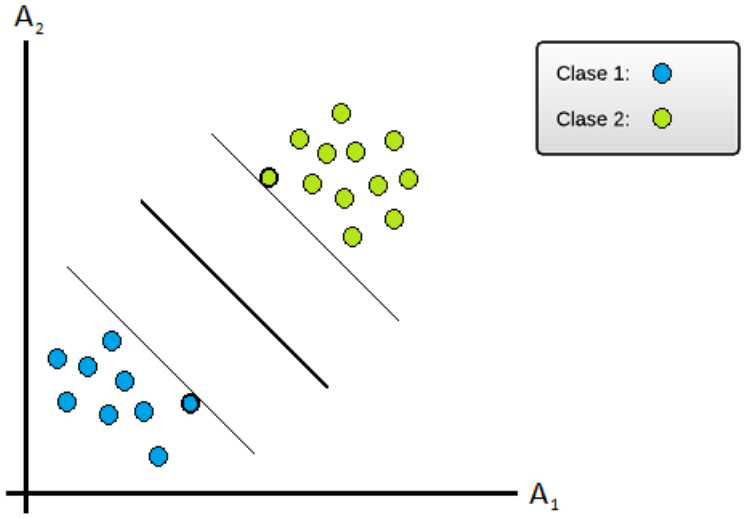

Figura 2.5: Hiperplano de margen máximo en método SVM
Fuente: Ejemplo adaptado de [15].

Como se aprecia en la Figura 2.5, el hiperplano de margen máximo se ubica entre los dos hiperplanos paralelos cercanos a los datos. La distancia entre ellos es optimizada, siendo estos dos hiperplanos paralelos quienes establecen las restricciones al problema de optimización. Todas aquellas tuplas de datos que se encuentren en estos hiperplanos paralelos se denominan *support vectors* (en la Figura 2.5 corresponden a los círculos con borde grueso). Por esto, todos ellos se encuentran a una misma distancia del hiperplano separador y son las tuplas de datos más difíciles de clasificar, y las más informativas respecto a la clasificación [15].

2.6 Twitter

Para poder aplicar las metodologías descritas anteriormente, es necesario contar con una fuente de datos lo suficientemente versátil y variada como para poder construir un conjunto de datos robusto que permita caracterizar el fenómeno que se pretende identificar. Las características de la red social *Twitter* facilitan esta construcción y permite utilizar éstas para generar datos previamente etiquetados bajo cierto criterio [32]. A continuación se describirán sus atributos principales.

2.6.1 Twitter y el concepto de Microblogging

Twitter corresponde a una red social que permite a sus usuarios publicar actualizaciones o estados en forma de textos cortos. Este tipo de servicios se denominan servicios de *Microblogging*, el cual corresponde a una forma de *blogging* (forma de comunicación basada en la Web, donde los usuarios a través de blogs publican contenido de distinto tipo según sus preferencias y de forma frecuente [33]) en el que los usuarios publican textos breves acerca de sus intereses o vida cotidiana y los envían a sus amigos u observadores interesados en el contenido, mediante mensajes de texto, mensajes instantáneos, correo o en la Web. Los tópicos expresados a través de Twitter varían desde situaciones de la vida cotidiana hasta

temas de contingencia, noticias locales o internacionales y otros intereses [34].

2.6.2 Usuarios

Twitter es utilizado por millones de personas. Desde tener un poco más de 90.000 usuarios a meses de su lanzamiento en 2006, ha llegado a tener más de 280 millones de usuarios activos en 2014 [34, 35]. Un usuario puede ser representado por cualquier persona o cualquier cosa [36]. Existen cuentas de Twitter que representan no solo a personas si no que organizaciones públicas o gubernamentales, empresas, emprendimientos grandes y pequeños, personajes ficticios e incluso cuentas a modo de parodia, basadas en personajes públicos. Los usuarios pueden decidir qué contenido ver o quienes desean que vean el contenido que generan. Para esto, los usuarios pueden seguir a otros y ser seguidos por otros. Esta relación no necesariamente es recíproca, si no que queda a opción de los mismos usuarios. Los usuarios mantienen un perfil personal, donde incluyen datos como nombre completo, ubicación, página web y una pequeña reseña o biografía sobre él. Este perfil también incluye quienes siguen a un usuario en particular, y a quienes sigue este usuario [37]. Mediante el contenido que los usuarios generan, los usuarios pueden mencionarse unos a otros, estableciendo una comunicación directa o también pueden hacerlo de forma privada mediante mensajes personales [38].

2.6.3 Contenido

Como se ha mencionado previamente, los usuarios publican mensajes cortos. Estos se denominan actualizaciones de estado (*status*) o de forma simple *tweets* y tienen un límite de 140 caracteres. Un tweet puede cumplir distintos roles en función de su naturaleza. Ésta es caracterizada por las acciones que le dieron origen y por el tipo de contenido que incluyen. A continuación se describen estos roles:

- **Status:** Corresponde simplemente a la publicación de contenido. Este puede incluir texto, fotos, video o enlaces a sitios externos.
- **Retweets:** Los usuarios pueden replicar o citar los mensajes de otros usuarios mediante un *retweet*. Los usuarios de un tweet que ha sido retwitteado es notificado, y el tweet original almacena esta información, manteniendo un conteo de las veces que ha sido retwitteado.
- **Menciones:** Como se menciona en la descripción de usuarios, una de las formas que tienen para comunicarse entre ellos es mediante *menciones*. Estas se realizan añadiendo al tweet el nombre de usuario que se desea mencionar anteponiendo un @ (arroba).

Además, los tweets de un usuario pueden ser marcados como favoritos. De forma similar al retweet, los tweets almacenan esta información y mantienen un conteo de la cantidad de veces que han sido marcados como favoritos, así como quienes son los usuarios que realizaron esta acción. Los usuarios que crean un tweet también tienen la atribución de poder eliminarlos.

Finalmente, en Twitter existe la posibilidad de indexar los tweets en función de su contenido. Esto se realiza mediante el uso de *hashtags*, los cuales se crean anteponiendo un

(símbolo numeral) a cualquier cadena de caracteres. De preferencia se utilizan palabras o frases incompletas que puedan ser complementadas en torno a una situación común, por ejemplo:

- #Tuitea2Palabras: Este hashtag contiene una frase que si bien no es del todo incompleta, es compartida con el fin de que los usuarios que lo utilicen sigan el sentido transmitido por éste. En este caso, se le pide a los usuarios que en sus tweets solo incluyan dos palabras de su preferencia.
- #SiElViejoPascueroFueraChilensis: Este ejemplo si incluye una frase incompleta, para que sea completada por los usuarios que lo utilicen.

Esta indexación mediante tópicos facilita la búsqueda de contenidos a los usuarios de Twitter, pero además facilita la categorización del contenido que los usuarios generan. En el capítulo 3 se verá que esta característica de Twitter fue fundamental para poder utilizar sus datos como fuente para la construcción de un corpus pensado en la detección de ironía.

2.6.4 Trending Topics

En Twitter, las palabras, frases y hashtags utilizados, citados y mencionados con mayor frecuencia son categorizados temporalmente como *Trending Topics* o TT. Estos se muestran en todo momento a los usuarios, mediante un *top ten* en la página principal. Los TT no son agrupados según el tópico al cual hacen referencia. Por esta razón es muy común que ante una situación que llame la atención de forma masiva, existan distintos TT asociados a un mismo tema.

2.7 Ironía

Como se ha mencionado, la presencia de ironía en el contenido generado por usuarios de Twitter y la Web en general, representa un problema para la aplicación de Análisis de Sentimientos debido a su naturaleza subjetiva. Por esta razón, es de interés de este campo realizar una caracterización de este fenómeno con el objetivo de que sea identificable utilizando las técnicas actuales de categorización de texto. Para ello, es importante realizar una revisión de aquellos elementos que definen este fenómeno como tal.

2.7.1 Lenguaje Figurado

Al utilizar el lenguaje para comunicarse entre personas, existen otros factores en el proceso comunicativo que influyen en el significado del mensaje que se emite, el cual puede divergir del significado directo de las palabras que lo conforman. Por esto, es posible diferenciar el *lenguaje literal*, donde los mensajes transmiten exactamente el significado literal que construyen sus palabras, y el *lenguaje figurado* el cual involucra un significado indirecto, que requiere un análisis más profundo para su interpretación. Existen diversas formas de lenguaje figurado, como la metáfora, la metonimia, el hipérbole, etc. y entre ellos se incluye la *ironía* ya que es una figura retórica que incluye un significado distinto al literal.

2.7.2 Definición de Ironía

La teoría tradicional define la ironía como una figura retórica que consiste en dar a entender lo contrario de lo que se dice, o sea, su significado es contrario al significado literal de su mensaje. A partir de esto, podría parecer que es posible dar con soluciones simples y directas para caracterizar este fenómeno, sin embargo, la evidencia empírica muestra que esto no es tan sencillo ya que generalmente no es claro este significado opuesto e incluso a veces no es claro el significado literal del mensaje. Debido a esto, el concepto de ironía ha sido definido desde diversos puntos de vista [39].

Clark et al. (1984) en [40] analiza dos teorías que buscan explicar los fundamentos de un mensaje irónico. Una corresponde a *teoría de la mención* en la que se plantea que la ironía puede ser explicada a través de la diferencia que existe en el *uso* y la *mención* de cierta expresión. Al ser irónico, un hablante no busca utilizar las palabras de una frase como si fueran propias, si no que busca solo mencionarlas haciendo referencia a otras frases (generalmente dichas por otra persona) que pueden formar parte de sabiduría popular u opiniones aceptadas, con el fin de hacer notar desprecio hacia estas o de indicar lo opuesto a lo que se dice. Veamos una adaptación en Español del ejemplo que cita [40]:

- *Confía en la oficina meteorológica. Mira el maravilloso clima que hay: lluvia, lluvia, lluvia.*

La teoría de la mención asume que el hablante no utiliza la frase *mira el maravilloso clima que hay* ya que el no busca que quienes le escuchan piensen que el clima es maravilloso, si no que solo menciona la frase, aludiendo a lo que podría decir un meteorólogo, con el fin de hacer entender lo contrario.

La segunda teoría corresponde a la *teoría de la pretensión* en la que se plantea que al expresarse de forma irónica, el hablante pretende ser reconocido o percibido de cierta forma (sin dar a conocer esta pretensión para que no se pierda el efecto en el oyente). Esta teoría incluye algunas características fundamentales de la ironía como:

1. En general, la ironía corresponde a una pretensión positiva sobre algo negativo. O sea, el hablante dice algo con un sentido positivo para criticar algo negativo. Este caso es más frecuente que el contrario dada la tendencia de las personas a ver el mundo basados en normas de éxito.
2. Suelen ser dirigidas a alguien, por lo que se dice que tiene *víctimas*. Estas son primero quien el hablante pretende ser (a quien replican o imitan sus palabras y/o actitudes) y segundo, los oyentes que reciben este mensaje en la ignorancia.
3. En su pretensión, el hablante no solo emite un mensaje sino que también le da un tono de voz adecuado, con el fin de exagerar o caricaturizar lo que pretende ser o representar.

La teoría de la pretensión es aplicable a muchas expresiones de ironía en las que también concuerda la teoría de la mención, pero esta relación no se da en la otra dirección, por lo que los autores señalan que la teoría de la pretensión permite interpretaciones mucho

más versátiles. Para concluir, se señala un factor psicológico que incide en la expresión de ironía. Los mensajes son irónicos solo para algunos oyentes ya que la percepción de ironía es dependiente de ciertos juicios sutiles en base a cierta información que el hablante y los oyentes/lectores tienen en común, por lo que si un a un oyente o lector no se le provee la información adecuada, no podrá emitir estos juicios correctamente. En otras palabras, si el hablante no es capaz de otorgar la información necesaria para que quienes reciben su mensaje puedan descubrir su pretensión, el mensaje no logra ser interpretado como irónico.

Junto a estas interpretaciones, en la literatura es posible encontrar dos tipos de ironía, pero a pesar de ello las definiciones para éstas no son simples. Una de éstas es la *ironía verbal*, la cual es definida tradicionalmente como un tropo (una expresión en la que se utiliza el sentido figurado o metafórico) en el que la expresión transmite un significado opuesto al significado literal de sus palabras. El problema de esta definición es analizado por Wilson et al. (1992) [41] recalcando que ésta no es total ya que es posible encontrar múltiples contraejemplos, como el uso de la ironía como una subestimación exagerada, como una citación o como una interjección (las cuales no son una oposición) e incluso como una falsedad o algo contrario que podría cumplir con esta definición tradicional pero dado el contexto, no sería reconocida como una ironía por los oyentes o receptores del mensaje.

Por esto, Wilson propone una extensión de la teoría de la mención diciendo que la ironía verbal puede ser vista como una variedad de expresión que *hace eco* o que referencia la proposición de otro hablante, para poder expresar cierta actitud hacia esta proposición citada. Con esto, añade que la ironía verbal involucra una actitud de desaprobación hacia esta proposición citada, usando desde el ridículo o burla leve, hasta un desprecio total. Para finalizar, añade que en todos los casos de ironía se hace eco o se referencia otra opinión u frase y que al mismo tiempo el hablante se disocia de ésta (se desliga o la critica), generalmente porque cree que es falsa (y termina expresando lo opuesto) o porque es demasiado absurda.

Otro tipo de ironía citado por la literatura corresponde a la *ironía situacional*. Lucariello en [42] la define como una condición de eventos opuestos respecto a lo que se espera naturalmente, lo que resulta en una contradicción. En general son eventos que se burlan del orden normal de las cosas. A pesar de que no son exactamente el mismo fenómeno, la ironía situacional se relaciona con la ironía verbal en que ambas son una oposición de términos y, además, ambas comparten un factor inesperado o sorpresa.

2.7.3 Rol en la comunicación

Más allá de sus diversas interpretaciones, la ironía es frecuentemente utilizada en distintas instancias de comunicación y es relevante a la hora de estudiar opiniones o análisis críticos extraídos desde internet. Esto se le atribuye a sus funciones sociales y al rol que tiene en la comunicación. Dews et al. (1995) [43] hace una revisión de las distintas funciones sociales de la ironía, partiendo desde la base de que existen instancias recurrentes en que las personas no dicen directamente lo que quieren decir y para ello, la ironía es usada ámpliamente como una herramienta de lenguaje no literal. Señala que la forma más común es la *crítica irónica* en la que un hablante dice algo positivo para transmitir algo negativo. Las funciones sociales de la ironía son:

1. Humor: Para los hablantes, puede serles útil ser divertidos mientras son críticos. La ironía tiende a ser más divertida que el lenguaje literal al momento de criticar, debido al factor sorpresa que conlleva.
2. Elevación de estatus: Un hablante que critica a otro, suele subir su estatus respecto a éste. Este efecto puede ser influenciado por la ironía de dos formas. Podría potenciar el efecto el hecho de criticar irónicamente ya que mientras critica, se suele contrastar el comportamiento real del hablante criticado, versus el comportamiento que debió haber tenido. Alternativamente, podría no elevar el estatus si la ironía se confunde con una broma.
3. Agresión: La ironía suele ser descrita como una forma ruda de criticar, que podría ser incluso más insultante que una crítica literal y directa. Suele ser usada para burlarse y humillar a la víctima de la ironía. Sim embargo, se dice que la crítica irónica es menos agresiva que una crítica literal ya que reduce la posibilidad de confrontación y ,por ende, la posibilidad de futuros conflictos.
4. Control emocional: En base a las características señaladas anteriormente, en [43] se muestra que la ironía permite a las personas que critican tener cierto autocontrol, ser menos agresivos. Permiten que un crítico enojado pueda controlar sus expresiones y, a su vez, permite que los hablantes criticados se sientan menos insultados, lo que es menos dañino para la relación entre ambas partes.

Junto a estas funciones, en [44] se realiza un análisis de la ironía utilizada entre amigos, estudiando la comunicación entre un conjunto de alumnos universitarios. Entre las conclusiones interesantes encontradas en el estudio, se puede mencionar que en general un mensaje irónico es respondido por el hablante aludido con otra respuesta irónica, llegando casi a ser un *estado mental* de comunicación entre los hablantes.

2.8 Detección de ironía presente en texto

Gibbs en [44] señala elementos que dificultan la interpretación de la ironía, al recalcar que existen muchas expresiones irónicas que no encajan dentro de las teorías de pretensión o de mención, y que la entonación del hablante al momento de expresarse juega un rol clave en la comprensión por parte de un oyente. A su vez, en [41] se recalca que al tratar de reconocer ironía siempre existe riesgo de cometer un error, ya que las intenciones del comunicador no pueden ser decodificadas, sino que deben ser inferidas. Junto con lo mencionado anteriormente respecto a las diversas interpretaciones a nivel teórico que es posible hacer del fenómeno, se concluye que no existe una panacea o *receta* que permita detectar con total precisión la presencia de ironía en un texto escrito.

En la actualidad, no existen métodos computacionales específicos para la detección de ironía en texto [7] y, junto con esto, no existen muchos estudios en el campo del Análisis de Sentimientos que hayan buscado una solución (menos en Español). La mayoría de los intentos se han realizado por medio de una clasificación de texto en base a la identificación de atributos reconocibles en el texto. A continuación, se realiza una breve revisión de algunos estudios relacionados junto a sus resultados.

2.8.1 Procedimientos Actuales

A pesar de la complejidad asociada al problema, distintos autores han definido sus propios criterios para clasificar y diferenciar texto irónico y no irónico. La mayoría de los trabajos de investigación utilizan la definición tradicional de ironía verbal para caracterizarla.

- Carvallo et al. (2009) en [45] realiza una clasificación basada en distintos atributos, con el objetivo de clasificar ironía en idioma portugués. Entre ellos es posible identificar diminutivos, pronombres demostrativos, interjecciones, la morfología de los verbos, *cross-constructions*, todos estos pensados en detectar características muy ligadas al idioma portugués y también utilizó otros atributos más generales como el uso de puntuaciones, uso de comillas y expresiones de risa. Para la construcción de su corpus, utilizó posts escritos por usuarios en la página web de un medio de prensa portugués, reuniendo cerca de 1 millón de sentencias. Como cantidad estimada, los autores señalan que su corpus contenía cerca de un 10% de sentencias efectivamente irónicas. Al realizar la extracción de atributos, tuvo una baja cobertura con sus atributos (coincidencias cerca del 0.2% del millón de sentencias) por lo que solo conservó aquellos atributos con una frecuencia mayor a 100 en las distintas sentencias, descartando los atributos de diminutivos, pronombres demostrativos, morfología de los verbos y *cross-constructions*. Usando los atributos restantes, sus resultados mostraron valores de precisión mayores al 45% (más alto que su línea de base de 10% estimada en base a la muestra) siendo los atributos más representativos, el uso de expresiones de risas (emoticonos, onomatopeyas, etc.) y el uso de comillas.
- Davidov et al. (2010) en [46] busca reconocer sentencias sarcásticas usando datos extraídos desde Twitter y Amazon. Su definición de sarcasmo es casi idéntica a la definición tradicional de ironía. Utiliza un método semi-supervisado en el que extrae atributos de datos previamente etiquetados con un nivel de sarcasmo (del 1 al 5) y luego construye un vector de atributos en base a aquellos relevantes. Con esto realiza una clasificación respecto a los 5 niveles de sarcasmo. Su algoritmo de extracción reconoce palabras de alta frecuencia (HFW) y aquellas que representan mayor contenido informativo (CW) y las reemplaza por un marcador. Para identificarlas, busca aquellas palabras que sobrepasen una brecha de frecuencias, establecidas por los valores F_H y F_C para HFW y CW respectivamente. Finalmente, estos atributos son filtrados y son descartados aquellos que surgen en base a contenido genérico y que no aporte demasiada información. Complementando estos atributos, también se utilizan atributos sintácticos y algo más genéricos como el largo de las sentencias (en palabras), la cantidad de signos de exclamación, signos de pregunta, comillas y palabras en mayúsculas. Luego, sus atributos son caracterizados por secuencias ordenadas según estos marcadores. Para realizar la clasificación, se utiliza el método de los k-vecinos más cercanos.
- Reyes et al. (2011) en [47] construye un modelo que busca caracterizar la ironía desde un enfoque completamente lingüístico identificando secuencias de N-grams, POS N-grams (cadenas de elementos Part-of-Speech), y perfiles positivos o negativos, afectivos y divertidos en base a las palabras utilizadas en el texto, apoyándose en la construcción de un corpus que represente aquellos perfiles. Se utilizaron evaluaciones de clientes en Amazon para generar un set de datos irónicos, el cual se compone de aproximadamente 2.800 documentos. Para construir los sets no irónicos se utilizan datos de Amazon y

Slashdot.com reuniendo 8.000 documentos aproximadamente. La clasificación se realiza utilizando métodos de Naive Bayes, Support Vector Machines y Árboles de Decisión obteniendo resultados con niveles de *accuracy* sobre el 70%.

- González-Ibañez et al. (2011) en [48] enfoca su estudio en la red social Twitter y busca identificar la presencia de sarcasmo en base a atributos léxicos basados en diccionarios, y atributos pragmáticos como presencia de emoticonos y referencias. El corpus es creado en función de las anotaciones que los mismos usuarios realizan, utilizando *Hashtags* generando un corpus de aproximadamente 900 tweets por las siguientes categorías: sarcasmo, positivo y negativo. Utilizando Support Vector Machines y Regresión Logística para la clasificación, obtiene resultados que oscilan entre el 60% y 70% de *accuracy* dependiendo de las categorías de datos utilizadas para la clasificación.
- Finalmente Reyes et al. (2013) en [5] propone un modelo multidimensional, donde se caracteriza la ironía en función distintas dimensiones que se componen de una serie de atributos. Este trabajo es desarrollado utilizando este modelo como base por lo que se describirá a fondo en el Capítulo 4.

2.9 Análisis y Evaluación de Resultados

2.9.1 Métricas de Desempeño

Para analizar los resultados de un clasificador, existen métricas que permiten obtener qué tan bueno es el desempeño del clasificador al momento de predecir la etiqueta de cierta clase de datos. Para generalizar, se hace referencia a los sets positivos y negativos descritos anteriormente. Recordemos que los datos en los sets positivos son aquellos que nos interesa etiquetar o clasificar, mientras que todos los otros datos que no son de interés para la clasificación, forman parte del set negativo.

Con estos términos es posible describir cuatro conceptos básicos, que nos permiten generar métricas de precisión para los clasificadores. Éstas son descritas a continuación [15]:

- Verdaderos positivos (VP): Son aquellos datos del set positivo, que son etiquetados correctamente como positivos por el clasificador.
- Verdaderos negativos (VN): Son aquellos datos del set negativo, que son etiquetados correctamente como negativos por el clasificador.
- Falsos positivos (FP): Son aquellos datos del set negativo, que son etiquetados de forma errónea por el clasificador. Es decir, estos datos son en realidad del set negativo, pero fueron etiquetados como positivos.
- Falsos negativos (FN): Son aquellos datos del set negativo, que son etiquetados de forma errónea por el clasificador. Es decir, estos datos son en realidad del set positivo, pero fueron etiquetados como negativos.

Estos datos pueden ser resumidos en una tabla llamada *matriz de confusión*, la cual permite ver a grandes rasgos qué tan bien se está comportando el clasificador al reconocer

los datos pertenecientes a las distintas clases al mostrar cómo éstos han sido clasificados. En la Figura 2.1 se muestra un ejemplo de dos clases en el que la clase de interés corresponde a *Si* y las otras clases corresponden a *No*.

		Clases Predecidas	
		Si	No
Clases Reales	Si	VP	FN
	No	FP	VN

Tabla 2.1: Ejemplo de Matriz de Confusión.
Fuente: Elaboración Propia

Ahora, teniendo estas definiciones básicas es posible caracterizar las medidas que nos permiten evaluar el desempeño del clasificador. A continuación se describen las más significativas para el análisis [15]:

- Accuracy: Esta medida representa el porcentaje de datos que son etiquetados correctamente por el clasificador, como se aprecia en la Ecuación 2.2. Refleja qué tan bien el clasificador reconoce los datos pertenecientes a las distintas clases. Esta medida por si sola no es suficiente ya algunos problemas de clasificación son susceptibles a un *desequilibrio de clases* en las que una clase es poco frecuente con respecto a la otra, por lo que analizar solo esta medida podría dar una falsa percepción de que el clasificador tiene un buen desempeño, cuando en realidad lo que sucede es que una clase representa una mayor cantidad de los datos con los que se está probando el clasificador.

$$\frac{VP + VN}{VP + VN + FP + FN} \tag{2.2}$$

- Precision: Corresponde a una medida de exactitud, que refleja el porcentaje de datos positivos que fueron etiquetados correctamente. Su cálculo se muestra en la Ecuación 2.3 donde se comparan los verdaderos positivos con todos los datos etiquetados como positivos por el clasificador.

$$\frac{VP}{VP + FP} \tag{2.3}$$

- Recall: Es una medida que representa qué tan completa fue la clasificación. Refleja el porcentaje de datos positivos que fueron clasificados como tal. Como se ve en la Ecuación 2.4, se obtiene la cantidad que representan los verdaderos positivos respecto al total de datos positivos reales.

$$\frac{VP}{VP + FN} = \frac{VP}{P} \tag{2.4}$$

- F-measure: Esta medida combina a *precision* y *recall* en una sola. Se define como se aprecia en la Ecuación 2.5, donde β es un número real no negativo. Corresponde a la *media armónica* entre la medida de precision y recall (medida utilizada cuando se quiere obtener el promedio entre tasas). Generalmente se asigna el mismo peso para ambas medidas, definiendo $\beta = 1$.

$$\frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall} \tag{2.5}$$

2.9.2 K-fold Cross Validation

Cuando se entrena el algoritmo, generalmente los sets de entrenamiento y prueba son escogido al azar. Esto podría provocar sesgo en los resultados, por lo que para realizar una evaluación que reduzca este sesgo existen distintas metodologías. Una de las metodologías más frecuentadas corresponde a *K-fold Cross Validation* en la que el set completo de datos es dividido en k sub-sets de tamaño similar. Luego, el clasificador es entrenado y probado durante k iteraciones, y en cada una se aparta uno de los k sub-sets y se utiliza como set de pruebas, mientras todo el resto se utiliza como set de entrenamiento [31].

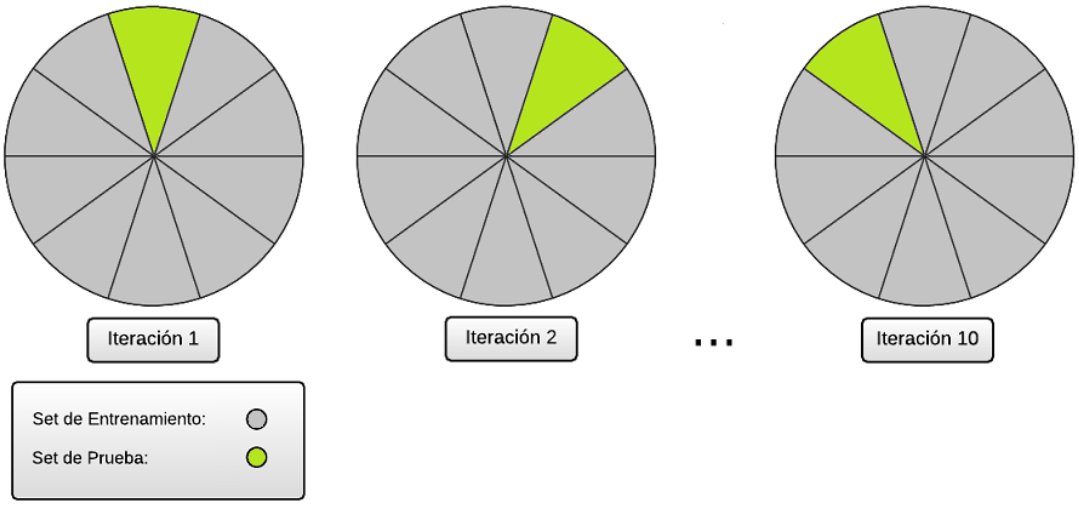

Figura 2.6: Ejemplo de 10-fold Cross Validation, uno de los casos más utilizados.
Fuente: Ejemplo adaptado de [31].

Capítulo 3

Construcción del corpus

El siguiente capítulo comprende la construcción del corpus a utilizar para la extracción de atributos que permitan caracterizar la presencia de ironía en el texto extraído desde Twitter. Se dará una definición del concepto de corpus y la importancia que tiene para la clasificación. Se describirá la metodología y los criterios utilizados en su construcción y se presentarán las herramientas tecnológicas empleadas para ello.

3.1 Definición

Un *corpus* corresponde a un conjunto estructurado de textos almacenados en forma electrónica y agrupados con el fin de estudiar fenómenos particulares del lenguaje. En el contexto de *Opinion Mining*, *Text Mining*, o de cualquier procedimiento que implique identificar patrones en texto, toma una gran relevancia ya que representa la herramienta a utilizar para identificar los patrones que hacen posible uso de algoritmos de aprendizaje supervisado. En esta ocasión, el interés está en construir un corpus en lenguaje Español que permita identificar patrones propios de la ironía utilizando herramientas de *Data Mining* y Procesamiento de Lenguaje Natural.

La ironía es uno de los muchos recursos lingüísticos del lenguaje figurado, el cual se caracteriza por la dificultad que involucra su procesamiento computacional ya que a diferencia del lenguaje literal, su significado no puede ser inferido directamente desde el significado literal del mensaje transmitido. El lenguaje figurado suele transmitir un significado subyacente. Por lo tanto, en este caso la construcción de un corpus lingüístico sólido es fundamental para el posterior proceso de identificación de patrones y clasificación.

3.2 Corpus para la detección de ironía

Un detalle importante para la construcción del corpus es que éste sea lo suficientemente completo para que conforme los datos de entrenamiento con los que se realizará el aprendizaje del algoritmo supervisado de clasificación. Para esto, los datos requieren estar previamente etiquetados en función del fenómeno que se quiere identificar o caracterizar a partir de éstos.

Pero además, es necesario contar con una gran cantidad de datos para que el conjunto de datos sea robusto y permita generar los patrones necesarios para la clasificación de nuevos datos ajenos al set de entrenamiento. Considerando esta situación, etiquetar manualmente los datos puede resultar muy difícil debido a la gran cantidad de datos que deben ser analizados. Por lo tanto, se aprovechará la existencia de *hashtags* en Twitter para realizar una recolección rápida de datos que además tengan relación con la ironía verbal.

3.2.1 Hashtags

En Twitter, los usuarios pueden crear contenido en forma de *tweets*, correspondientes a mensajes de 140 caracteres como máximo. Estos, a su vez, pueden incluir contenido complementario o alternativo al mensaje escrito como enlaces a otras páginas, imágenes, menciones a otros usuarios, etc. Además, los usuarios pueden categorizar sus tweets en función de distintos contenidos o tópicos utilizando *hashtags*. Un hashtag es una etiqueta construida por una cadena de caracteres, que generalmente se compone de una o más palabras sin espaciar, a la que se le antepone un símbolo numeral o #. La idea es que las palabras que componen un hashtag correspondan a palabras clave o frases relevantes para el tema en discusión. Si un tweet incluye un hashtag, éste quedará categorizado en torno al tópico caracterizado por éste. Por lo tanto, los usuarios de Twitter pueden utilizar los hashtag para indicar que sus tweets se relacionan directamente con algún tema en particular. Esto facilita que los tweets de un usuario se hagan presentes en la búsqueda de Twitter e incluso que se vuelvan populares al ser utilizados por más personas que desean generar más contenido en torno a ese tópico. Los hashtags más populares pueden transformarse en *Trending Topics*, permitiendo que otros usuarios puedan percibir el uso de éstos [49].

Por ejemplo, algunos de los *Trending Topics* para Chile correspondientes al día 17 de octubre del 2014 contienen los siguientes hashtags:

- #AhoraNoMeVendriaMal
- #MiNombreEs
- #LadyGaga

Si un usuario hace click en alguno de ellos, puede ver todos los tweets recientes asociados al tema. Por ejemplo, dentro de los tweets asociados al hashtag #MiNombreEs se puede encontrar:

1. Amy Winehouse y Lady Gaga son las últimas clasificadas en la categoría adultos para la Gran Final de #MiNombreEs
2. Amigos, ya estamos los 10 finalistas de #MiNombreEs Próximo jueves será la gran final y el jurado serán ustedes! Seamos más! #FrankSinatra

Se aprecia que los tweets asociados a este hashtag se relacionan con el programa de televisión "Mi Nombre Es", concurso de imitación de artistas transmitido durante esta temporada por Canal 13 de Chile.

3.2.2 Uso de Hashtags para recolectar datos

Considerando las facilidades que otorga la existencia de hashtags para la categorización de contenidos en Twitter, se utilizará esta herramienta para recolectar datos que permitan construir el corpus deseado. Se utilizarán dos hashtags que permitan caracterizar aquel contenido generado por los usuarios que sea potencialmente irónico:

1. #ironia
2. #ironico

Nótese que para efectos de búsqueda de contenido, estos hashtags incluyen símiles como #ironía, #Ironia, #Ironía, #irónico, #Ironico, #Irónico, etc. Se pretende recolectar una gran cantidad de tweets asociados a éstos. Más allá de las características de los hashtags, los fundamentos para utilizar este criterio tienen que ver con la intención comunicativa del autor de un mensaje.

Una de las dificultades asociadas a la construcción de un corpus pensado en la detección de la presencia de ironía tiene que ver con la diferencia semántica entre una expresión de lenguaje literal y una expresión de lenguaje figurado. Sin mayor información, a nivel textual un par de expresiones no son más que secuencias de palabras con cierto significado semántico. Por lo tanto, una forma de hacer sobresalir las diferencias entre ambas expresiones, tiene que ver con la intención comunicativa que haya tenido su creador, o sea, qué pretendía transmitir el autor de un mensaje. Esta intención comunicativa se relaciona con la necesidad del hablante o creador de la expresión, de escoger ciertas palabras o estructuras semánticas para transmitir con éxito el significado que éste pretendía comunicar (en otras palabras, cumplir sus intenciones). Citando los ejemplos de [32], consideremos las siguientes expresiones asociadas a un *arcoíris*:

- "Un arcoíris es un arco de luces de colores en el cielo causado por la refracción de los rayos de luz solar a causa la lluvia."
- "Un arcoíris es una promesa en el cielo."

Se aprecia que en el primer ejemplo se intenta describir qué es un arcoíris. En cambio, en el segundo ejemplo se muestra un significado subyacente, comprensible desde un contexto conceptual específico. En cada expresión, el hablante tiene una necesidad comunicativa, la cual logra al resaltar ciertos elementos; mientras que en la primera expresión el éxito de la comunicación se alcanza al hacer una precisa descripción de lo que es un arcoíris, en la segunda se basa en escoger deliberadamente elementos cuyas relaciones son secundarias y no literales.

En síntesis, si somos capaces de capturar esta intención comunicativa, se podría estar más cerca del significado real que el autor de un mensaje buscaba transmitir. Para esto, Reyes et al. (2012) en [32] propone utilizar hashtags para recolectar información desde Twitter, asociada a un mensaje irónico. Con esto, se está recolectando contenido previamente etiquetado como ironía por parte de sus autores, lo que hace más rápida y fidedigna la

construcción del corpus.

3.3 Método de Extracción

3.3.1 Fuentes Utilizadas

Para proceder a la extracción de los datos, se utilizaron dos fuentes. La primera fue directamente Twitter, por medio de una conexión a su *Streaming API*, que permite recibir en tiempo real parte del flujo generado de tweets filtrado en función de uno o más parámetros. Como el contenido obtenido a través de esto es generado en el momento, no es suficiente para recolectar una gran cantidad de tweets ya que los hashtags asociados a ironía son utilizados de forma espontánea por los distintos usuarios a una baja frecuencia, por lo que se decidió recurrir a otros métodos. Twitter también cuenta con una *Search API* que por medio de consultas, permite realizar búsquedas para recolectar tweets según ciertos parámetros, pero no puede entregar resultados de más de una semana de antigüedad. Por lo tanto, se optó por una fuente alternativa y externa a Twitter.

Topsy es una compañia de análisis de redes sociales, emparentada de forma oficial con Twitter y que mantiene grandes cantidades de datos indexados desde distintas redes sociales. Respecto a Twitter, en la aplicación web *topsy.com* es posible encontrar tweets en función de palabras claves o hashtags generados en cualquier periodo de tiempo, incluso aquellos generados en los primeros años de existencia de la red social. Tomando esto en cuenta, se decidió hacer uso de Topsy también como una fuente de datos en función de los hashtags utilizados anteriormente.

Entre ambas fuentes fueron recolectados 39.053 tweets, representados por una cadena en formato *JSON*.

3.3.2 Módulo de Extracción de Datos

Para recolectar los datos fue desarrollado un módulo de extracción de datos, constituido por dos *crawlers* implementados mediante el lenguaje *Python* en su versión 2.7. El primero almacena los datos recolectados mediante una conexión por medio de la *Streaming API*. Para esto, se utilizó la librería *python-twitter*[1], la cual provee una interfaz en Python para la API de Twitter.

El segundo extrae los tweets mostrados a través de *topsy.com* directamente desde el código fuente de la página web. Para ello, se utilizó la librería *Selenium* que permite disponer de los recursos de un navegador web. Lo que se extrae desde el código fuente es el ID de cada tweet. Luego, se volvió a utilizar *python-twitter* para obtener los datos completos de cada tweet, en base a las ID recolectadas.

[1]https://github.com/bear/python-twitter

3.4 Análisis Exploratorio y Limpieza de Datos

Realizando un análisis exploratorio de los datos, es posible identificar distintos problemas asociados a su origen y al contexto en el que fueron generados. Estos problemas son recurrentes al trabajar con Twitter como es sugerido en [20] y afectan directamente a la interpretación que es posible hacer respecto a cada tweet extraído, por lo tanto no serán útiles para la extracción de patrones posterior. Por esta razón, es necesario realizar una limpieza inicial a los datos para poder trabajar con ellos. A continuación se describen aquellos problemas identificados y se propone una solución para estos.

3.4.1 Problemas identificados en los datos extraídos

Tweets sin información relevante para el estudio: En Twitter, los usuarios pueden generar contenido de diversas formas. En repetidas ocasiones, es posible encontrar tweets que no aporta ningún tipo de información relevante para la construcción de este corpus. Por ejemplo, los usuarios pueden escribir tweets con contenido insuficiente como para transmitir algún tipo de información o también compartir enlaces a otras páginas o imágenes, los cuales son elementos que no pueden ser utilizados en el análisis de texto. Por ejemplo:

1. #Ironia ..
2. Que #IRONICO https://t.co/ICD5LINRMO
3. Que coche mas feo.... #ironia http://t.co/Oj6mBZRKm3

En el primer ejemplo, el autor utilizó el hashtag #Ironia y solo añadió puntos suspensivos. Este tipo de contenido no entrega ningún tipo de información respecto al contexto en que el tweet fue generado. Los dos ejemplos restantes muestran cómo se complementa un mensaje con una imagen. Sin ver las imágenes compartidas por sus autores, es imposible captar el mensaje irónico que quisieron transmitir ya que parte de la información no es contenida explícitamente en el texto. Por esta razón, el contenido no es suficiente para el análisis.

Presencia de Retweets: Además de escribir sus propios tweets, los usuarios de Twitter pueden replicar el contenido generado por otros usuarios. El contenido generado de esta forma se denomina *retweet* y la acción que da origen a éste se llama *retwittear*. Éstos pueden estar presentes de dos formas:

a) Creados a partir del botón "retweet", con lo cual los usuarios replican el contenido deseado y lo comparten con sus seguidores.

b) Creados a partir de una cita. Para realizar una cita en Twitter, los usuarios copian el contenido que buscan citar, y lo añaden a un tweet añadiendo las letras *RT* para señalar que el contenido corresponde a un retweet y no a un contenido propio. En estos casos, los usuarios pueden añadir sus propios comentarios al retweet, aunque este contenido se ve limitado al largo del contenido retwitteado. Por ejemplo:

- RT @Maghero: Nicolás le reclama a la oposición que condenen la violencia y el paramilitarismo #Ironia ??? http://t.co/v0yxRIuIp3

El contenido retwitteado corresponde a contenido repetido, por lo tanto no aportan nueva información al análisis y podrían afectar negativamente al algoritmo de clasificación al repetir de forma innecesaria algunas oraciones.

Información contenida en el Hashtag: Como se ha mencionado previamente, en Twitter los usuarios pueden utilizar *hashtags* para categorizar su contenido. Un hashtag puede ser compuesto por cualquier cadena de caracteres antepuesta por un símbolo numeral. Por diversas razones, hay usuarios que construyen hashtags en base a diversas palabras, las cuales forman parte importante de la información que está siendo transmitida un tweet determinado. Por ejemplo:

1. Gracias por ese cariño que me teneis #Ironia #PorSiNoOsHabiaisDadoCuenta
2. Cada día me gustan mas los anuncios de la cadena tres #Ironia #MediaHoraDeAnuncios

Estos casos son difíciles de procesar ya que si se deseara separar las palabras de forma automática, resultaría complejo determinar donde comienza y termina cada palabra de la cadena. Además hay usuarios que solo utilizan letras minúsculas o mayúsculas al crear este tipo de hashtags, complicando aún más su eventual análisis automático.

Respuestas a otros usuarios: Los usuarios de Twitter pueden enviar respuestas entre ellos haciendo uso de las *menciones*. Para realizar una mención, los usuarios deben añadir en sus tweets el nombre de usuario al que desea hacer la mención, anteponiendo un arroba (@) a éste. En un tweet es posible hacer mención a todos los usuarios posibles, mientras sus nombres de usuario puedan ser escritos dentro del límite de 140 caracteres. Por ejemplo:

1. @americanista_mx @clubleonfc claaaaaaroooo NIÑOS TELEVISA #ironia
2. @Debbora95 #ironia
3. @BlasAuryn porq eres tan feo?? ?? #ironia

El problema de este tipo de contenido es que generalmente las menciones forman parte de una conversación, por lo que cada mensaje no contendrá la información completa para poder entender el contexto de un mensaje. Como se aprecia en los ejemplos anteriores, aquellos tweets forman parte de una conversación, por lo que probablemente su contenido depende de un mensaje anterior o de la relación propia entre los usuarios, por lo que es imposible interpretar correctamente cada mensaje sin disponer de la conversación completa o de información adicional.

Idioma: Por último, un problema presente fue el idioma de los tweets. El hashtag #ironia y sus derivados también es utilizado en idiomas como el Italiano y el Portugués. Por lo tanto,

como el criterio de recolección fue la utilización de éstos hashtags, varios tweets en idiomas aparte del Español fueron recolectados en el proceso.

3.4.2 Solución propuesta a problemas identificados

Para dar solución a los problemas expuestos, se utiliza un filtro automático que identifica elementos característicos de cada problema y se eliminan del corpus. Estos elementos se describen a continuación:

Tweets sin información relevante para el estudio: Para identificar estos casos, se detectaron todos aquellos tweets que incluyeran algún tipo de enlace externo. Los enlaces pueden indicar la presencia de imágenes o de sitios externos que complementan el mensaje. Respecto al contenido, se analiza el largo de cada texto para solo mantener aquellos que superen cierta brecha.

Presencia de Retweets: Existen dos formas para identificar retweets. La primera es analizando los atributos del tweet, contenidos en el *JSON* que entrega la API de Twitter. Específicamente la existencia de la llave *retweeted_status* indica si un tweet fue retwitteado o no. La segunda forma es identificar las letras *RT*, las cuales indican un retweet a modo de cita. Ambas características fueron filtradas, eliminando del corpus aquellos tweets que coincidieran con éstas.

Información contenida en el Hashtag: La presencia de otros hashtags aparte de #ironia y sus símiles, no aportan mayor información. Por lo tanto, se optó solo por contar con aquellos tweets que utilizaron #ironia y sus símiles como el hashtag principal en su contenido. Se descartaron los tweets que contuviesen otros hashtags aparte de éstos.

Respuestas a otros usuarios: Las menciones también pueden ser identificadas de dos formas. La primera es en base a los atributos contenidos en el *JSON* que conforma el tweet. La llave *in_reply_to_status_id* indica la *ID* del tweet (también llamado *status*) al cual se está respondiendo. Si ésta tiene un valor distinto de "None", quiere decir que efectivamente el tweet es una respuesta a otro y por lo tanto está realizando una mención. La segunda forma de identificar las menciones es buscando en el texto del tweet una o más arrobas, las cuales son utilizadas para construir las menciones como se explicó anteriormente.

Idioma: Para identificar exclusivamente aquellos tweets escritos en Español, también se hizo uso de los atributos de éstos, definidos gracias al formato *JSON*. La llave *lang* indica el idioma del tweet en cuestión, por lo tanto, solo se mantuvieron aquellos que tuviesen esa llave con el valor "es".

Luego de la aplicación del filtro, se rescataron 17.277 tweets. Cabe mencionar que antes de aplicar estos filtros, se eliminó del texto el hashtag #ironia y sus símiles. Los resultados del procedimiento se aprecian en la tabla 3.1.

Descripción	Cantidad
Tweets analizados	39.053
Tweets descartados por ser irrelevante	1.673
Tweets descartados por ser retweet (llave)	4.125
Tweets descartados por ser retweet (cita)	6.908
Tweets descartados por ser respuesta o mención	8.090
Tweets descartados por lenguaje	1.947
Tweets efectivamente descartados	17.235
Tweets filtrados	17.277

Tabla 3.1: Resultados de filtrado automático de Tweets.
Fuente: Elaboración Propia

3.5 Clasificación Manual

Para complementar el criterio propuesto en [32], se utilizará la *sabiduría de las masas* para perfeccionar el contenido del corpus. La identificación automática de ironía en lenguaje natural siempre ha representado un desafío debido a las diversas definiciones e interpretaciones formales que es posible hacer respecto a este fenómeno. Sin embargo, como se menciona en [23], los investigadores concuerdan en que las personas son capaces de detectar la presencia de ironía en un texto de forma efectiva. Por lo tanto, una vez realizada la limpieza inicial de los datos, éstos se filtrarán utilizando clasificadores humanos para determinar cuales textos representan de mejor forma la expresión de ironía.

Los tweets filtrados se dividen en 10 sets de 1.728 textos aproximadamente. Luego, utilizando 20 evaluadores humanos[2] se revisa cada set y se asigna una categoría a cada uno de sus textos. Cada set es evaluado por dos personas distintas y desconocidas entre si, sin ningún tipo de comunicación entre ellas. Para evaluar, cada persona añade una etiqueta a cada tweet del set. La evaluación se realiza de la siguiente forma:

- Se le indica a los evaluadores que deben utilizar su propia intuición para detectar presencia de ironía en cada uno de los textos presentes en el set. La idea es que, al igual como los usuarios generan contenido irónico en redes sociales, la identificación sea basada principalmente en la intuición humana y no en una definición formal de ironía.
- Si el evaluador considera que el texto es irónico (basado en su intuición y en la información contenida en el texto), debe etiquetarlo con un "Y".
- Si el evaluador no detecta la presencia de ironía o si el texto no se entiende, debe etiquetarlo con un "N".

El criterio de selección se basa en la unanimidad de los evaluadores. Si ambos evaluadores escogen la misma etiqueta para un texto determinado, entonces aquel tweet es clasificado

[2]El grupo de evaluación se conformó por 20 personas, hombres y mujeres en un rango etareo de 20 a 50 años, con un nivel educacional equivalente a educación superior, entre los que se encuentran estudiantes, egresados y profesionales titulados. No se exigió ningún criterio en especial para participar de la evaluación.

según su etiqueta ("Y" para contenido irónico y "N" para contenido no irónico). Si los evaluadores no concuerdan con su decisión, entonces el texto es revisado por un tercer evaluador y su decisión es la que determinará la categoría del tweet en cuestión. Con este criterio en consideración, la evaluación pasó por dos etapas antes de tener una evaluación final del total de tweets que conforman el corpus.

3.5.1 Primera Evaluación

Entonces, la primera evaluación da como resultados un conjunto de concordancias (donde ambos evaluadores concordaron en su evaluación) y discrepancias (donde los evaluadores no estuvieron de acuerdo en su evaluación). Del total de tweets evaluados, un 53.4% fueron parte de las concordancias y un 46.1% fueron discrepancias. En las concordancias, un 33.1% de los tweets fueron evaluados como positivos y un 20.2% como negativos. Estos resultados se resumen en la tabla 3.2.

Resultados Evaluación Inicial		
Concordancias	Positivos	5.728
	Negativos	3.579
Discrepancias	Total	7.970

Tabla 3.2: Resultados de evaluación inicial de tweets irónicos.
Fuente: Elaboración Propia

Para realizar un análisis descriptivo de esta evaluación, es posible utilizar la medida estadística *kappa de Cohen* [50]. Esta medida permite obtener una medida del grado de concordancia que tiene un par de evaluadores al etiquetar sujetos según un conjunto de clases. Como considera que los dos evaluadores etiquetan cada dato de la muestra, se obtendrá esta medida para cada uno de los 10 sets en los que se dividió la muestra completa. Para calcularla, se utilizan las proporciones en las que los evaluadores seleccionan una u otra clase para etiquetar las muestras, y como en este caso tenemos solo dos clases posibles (irónico o no irónico) para cada set tendremos una matriz de $2x2$ con las proporciones, como se aprecia en la tabla 3.3.

		Evaluador B	
		Si	No
Evaluador A	Si	a	b
	No	c	d

Tabla 3.3: Ejemplo de Matriz de proporciones utilizada para obtener el kappa de Cohen de un set.
Fuente: Elaboración Propia

Entonces sean a, b, c y d las proporciones que representan las discrepancias y coincidencias entre los evaluadores A y B de un set cualquiera, con N textos (como son proporciones, se cumple que $a + b + c + d = 1$). La medida *kappa* se calcula como muestra la Ecuación 3.1.

$$\kappa = \frac{P_0 - P_e}{1 - P_e} \tag{3.1}$$

Donde los valores P_0 corresponde al *acuerdo relativo observado* y se calcula como la suma de las proporciones en las que los evaluadores concuerdan en su etiqueta, mientras que P_e corresponde a la *probabilidad de acuerdo* y se calcula en base a las proporciones totales de decisión de cada evaluador. Así, de la tabla se observa que un evaluador A etiquetó como irónicos (caso Si) un $(a+b)\%$ de la muestra y como no irónicos (caso No) al $(c+d)\%$ restante. De la misma forma, el evaluador B etiquetó un $(a+c)\%$ y un $(b+d)\%$ como irónico y como no irónico respectivamente. Luego, la probabilidad de que algún evaluador etiquete algún dato de la muestra como irónico corresponde a $(a+b)*(a+c)$, y la probabilidad de etiquetar como no se define por $(c+d)*(b+d)$. Luego, la probabilidad de acuerdo al azar total se calcula como la suma de ambas probabilidades. El cálculo de P_0 y P_e se puede apreciar en las Ecuaciones 3.2 y 3.3 respectivamente.

$$P_0 = a + d \tag{3.2}$$

$$P_e = ((a+b)*(a+c)) + ((c+d)*(b+d)) \tag{3.3}$$

Los resultados del cálculo de *kappa* para cada set, se aprecian en la Tabla 3.4. Si los evaluadores estan completamente de acuerdo, $\kappa = 1$. Si el acuerdo observado es mayor o igual a la probabilidad de un acuerdo entre los evaluadores, $\kappa \geq 0$ y si el acuerdo observado es menor a la probabilidad de acuerdo, $\kappa \leq 0$. Ante esto, se podría decir que en todas las evaluaciones existe cierto nivel de concordancia, sin embargo en [50] además se plantea que en la literatura los valores aceptables suelen estar entre 0.4 y 0.75, por lo que ningún set tendría un valor de κ estadísticamente significativo. Esto evidencia la dificultad que tiene la detección de ironía a partir únicamente de texto, incluso para seres humanos. Para poder determinar una segunda opinión sobre aquellos etiquetados en los que los evaluadores no estuvieron de acuerdo, se realiza una segunda evaluación como se describe a continuación.

Cálculo de κ		
Set	κ	Error estándar
1	0,36	0,02
2	0,24	0,02
3	0,03	0,02
4	0,05	0,02
5	0,11	0,02
6	0,2	0,02
7	0,12	0,01
8	0,08	0,01
9	0,15	0,02
10	0,01	0,01

Tabla 3.4: Cálculo de medida kappa para cada set.
Fuente: Elaboración Propia.

3.5.2 Segunda Evaluación

Luego, la segunda evaluación correspondiente a las discrepancias dio como resultado un 51.2% de tweets positivos y un 48.8% de tweets negativos, como se aprecia en la tabla 3.5. En relación al total de tweets, esto representa un 23.6% y un 22.5% respectivamente.

Resultados Evaluación Discrepancias	
Positivos	4.084
Negativos	3.886

Tabla 3.5: Resultados de evaluación de discrepancias en la evaluación inicial.
Fuente: Elaboración Propia

Finalmente, en la tabla 3.6 se aprecian los resultados finales del proceso de evaluación completo. El corpus fue etiquetado manualmente con un 56.8% de tweets positivos y un 43.2% de tweets negativos. Es decir, de los 17.277 tweets recolectados y filtrados, un 56.8% fue evaluado como *ironía* y un 43.2% fue evaluado como *no ironía*. Con estas cantidades es posible extraer un set de entrenamiento que contenga cantidades similares de tweets con las distintas clases que interesa clasificar en este caso, junto con un set de prueba para verificar la efectividad del clasificador. Sin embargo, para diversificar las instancias de clasificación, se conformará otro set con casos negativos.

Resultado Final de Evaluación	
Positivos	9.812
Negativos	7.465

Tabla 3.6: Resultado final de evaluación manual de tweets.
Fuente: Elaboración Propia

3.6 Sets de Casos Negativos

Para realizar el entrenamiento del algoritmo de clasificación, y para probarlo posteriormente, es necesario contar con un set de datos que contenga todas las clases de datos que se pretende identificar. En este caso, interesa clasificar dos tipos de textos: textos irónicos (set positivo) y textos no irónicos (set negativo). En la sección anterior se describe el proceso que llevó a la construcción de un set positivo y uno negativo en función de evaluadores humanos, quienes asignaron una etiqueta a cada texto del corpus. Por lo tanto, ya se dispone de un set negativo para el proceso de entrenamiento y prueba.

Sin embargo, para diversificar las pruebas, se utilizará otro set negativo conformado de una forma diferente. Para su construcción se utilizará *texto objetivo* en los cuales se transmite información, sin contener una apreciación u opinión por parte de quien emite el mensaje. Este tipo de textos no contienen ironía ni otro tipo de lenguaje subjetivo.

Como fuentes de textos objetivos se utilizaron las cuentas de twitter de distintos portales de noticias de Chile. Estas corresponden a las mismas utilizadas por Tapia en [20] para recolectar texto objetivo bajo el supuesto utilizado por Pak y Paroubek (2010) en [51], el que

establece que generalmente los medios de prensa emiten sus noticias de forma objetiva, sin agregar juicios particulares. Éstas se resumen en la tabla 3.7.

Cuentas de Prensa	
Medio de Prensa	**Cuenta asociada**
Radio Bio Bio	@biobio
CNN Chile	@cnnchile
Radio Cooperativa	@cooperativa
Emol.com	@emol
Diario La Tercera	@latercera

Tabla 3.7: Cuentas de prensa utilizadas para la recolección de texto objetivo.
Fuente: Adaptada de [20]

Para la recolección se utilizó el crawler construido para *topsy.com* utilizados anteriormente, solo que utilizando como parámetro de búsqueda los tweets emitidos por las cuentas de prensa. Esto permitió recolectar 19.003 tweets con texto objetivo los cuales fueron sometidos a un procedimiento de filtrado automático similar que el aplicado a los tweets potencialmente irónicos. Este filtrado dio como resultado un set de 17.855 tweets con texto objetivo, los cuales son considerados como un set de datos negativo para el proceso de entrenamiento y prueba del algoritmo de clasificación.

Capítulo 4

Modelo de detección de Ironía y Método de Clasificación

Una vez que ya se tiene el set de entrenamiento para el algoritmo de clasificación, se debe abarcar la extracción de atributos para poder caracterizar las clases que se busca clasificar. En este capítulo se describe en detalle un modelo para la detección de ironía, basado en distintos atributos que pretenden abarcar distintos elementos del texto y también elementos asociados a las emociones de los autores del contenido. Además se presenta el algoritmo de clasificación que se utilizará posteriormente para la identificación automática de ironía.

4.1 Modelo de Detección de Ironía

El modelo que se utilizará corresponde a una adaptación del modelo "multidimensional" para la detección de ironía, desarrollado por Reyes et al. (2013) [5]. Es llamado así ya que se compone de 4 *macro-atributos* que tratan de capturar distintas características sintácticas y semánticas del texto analizado. Fue escogido ya que, se encuentra dentro del estado del arte de los métodos actuales para la detección automática de ironía y además en comparación con otros métodos desarrollados trata de llevar el análisis más allá del procesamiento de lenguaje natural, añadiendo elementos del ámbito del análisis de sentimientos e incluso la psicología.

4.1.1 Dimensiones

Como se ha mencionado, el modelo se compone de dimensiones o *macro-atributos*, los cuales no son más que conjuntos de atributos similares enmarcados en un concepto, tratando de capturar propiedades a distintos niveles. En total corresponden a 11 atributos distribuidos en las 4 dimensiones, cada uno con distintas características pensadas para capturar elementos puntuales que potencialmente pueden ser parte de la expresión de ironía en un texto. En la tabla 4.1 se resumen las dimensiones del modelo junto a sus respectivos atributos.

A continuación se describirá en forma breve cada uno de los atributos que componen las 4 dimensiones del modelo.

Dimensión	Atributos		
Signatures	Pointedness	Counter-factual	Temporal Compression
Unexpectedness	Contextual Imbalance	Temporal Imbalance	
Style	Character N-grams	Skipgrams	Polarity Skipgrams
Emotional Context	Activation	Imagery	Pleasantness

Tabla 4.1: Dimensiones del Modelo de Detección de Ironía junto a sus atributos.
Fuente: Elaboración Propia

- **Signatures:** Esta dimensión busca caracterizar elementos textuales, frecuentes en la ironía y que hagan énfasis en aspectos puntuales del texto analizado. Se basa principalmente en símbolos tipográficos y palabras seleccionadas previamente. Para la identificación de estos elementos, esta dimensión es compuesta por 3 atributos distintos:

  Pointedness: Este atributo busca capturar símbolos puntuales, que podrían estar eventualmente presentes en un texto donde se use ironía para expresarse. Estos símbolos corresponden a emoticonos, signos de puntuación y palabras en mayúsculas. Por ejemplo:

  - Mi autobus retrasado**???** **NOOO** me estan mintiendo verdad**??** Pero si eso **JAMAS** sucede ¬¬ **!!!**
  - Mañana de bautizo**...** Q ilusion**!!!** **xD**

  Los elementos que este atributo busca capturar se encuentran remarcados. Para identificar emoticonos, se utilizó una versión adaptada de la lista de emoticonos utilizada en [5] la cual fue complementada con otros tipos. Los signos de puntuación identificados corresponden a ¿, ?, ¡, !, ., ,, :, ;, y uso de comillas (¨). La identificación además considera cadenas repetidas con estos elementos, como por ejemplo *puntos suspensivos* (...).

  Counter-factuality: Este atributo se basa en detectar indicios de oposición o contradicción en el texto, mediante la identificación de adverbios escogidos de forma puntual, que pueden dar señales de una negación. La lista de adverbios utilizada fue una versión adaptada al idioma Español de la lista utilizada en [5] junto con sus sinónimos relacionados (también adverbios). También se incluyen términos compuestos por más de una palabra como *sin embargo*, *por lo tanto* o *más o menos*. A continuación se incluyen ejemplos de lo que se busca detectar:

  - Yo si te quiero... **pero** lejos, muy lejos de aquí.
  - Los profesores nos llaman desactualizados porque no usamos como debemos las computadoras, **sin embargo**, **solo** mandan trabajos a mano

  Las palabras remarcadas en los ejemplos anteriores forman parte de la lista de adverbios señalada.

  Temporal Compression: De forma similar al anterior atributo, *Temporal Compression* trata de identificar cambios abruptos en la narrativa mediante la detección de términos que indiquen oposición en el tiempo. Para esto, se utiliza una lista de adverbios temporales. Al igual que en el atributo anterior, se utiliza una versión adaptada al

Español de la lista propuesta en [5], junto con sus sinónimos relacionados. Al igual que en el atributo anterior, esta lista incluye términos compuestos por más de una palabra. Entre ellos están *en breve*, *desde entonces* o *dentro de poco*, entre otras. La detección sería como se muestra a continuación:

- Qué mejor para las matas de mi casa **después** de este verano tan duro que el aguacero que está cayendo **ahora**
- Me encanta cuando me juzgais sin saber mi actitud **desde** hace unos meses hasta **ahora**! ME ENCANTA.

Como se aprecia en los ejemplos, las palabras remarcadas forman parte de la lista de adverbios en la que se basa el atributo.

- **Unexpectedness:** Esta dimensión captura los elementos inesperados o desequilibrios en la narrativa del texto, propios de la expresión de ironía. Sus atributos están basados en el estudio desarrollado en [42], donde Lucariello propone una serie de elementos característicos de la ironía. Entre éstos se encuentran los "Imbalances" o desequilibrios, los cuales representan casos de oposiciones o inconsistencias en el comportamiento humano o en situaciones esperadas.

  Contextual Imbalance: Este atributo se basa en el grado de coherencia contextual que existe en el texto analizado. Para poder capturar esto, se utiliza una medida de similitud semántica entre las palabras que componen el texto. Esto entrega una medida, la cual entre mayor sea su valor, mayor es esta similitud. Como se ha mencionado previamente, en un texto en el que se trata de expresar ironía, uno espera encontrar elementos inesperados o fuera de un contexto común, por lo tanto [5] propone esta medida para caracterizar los eventuales desequilibrios en el contexto durante la emisión del mensaje contenido en el texto. Como a mayor valor de similitud semántica en el texto, mayor es la coherencia en el contexto, [5] plantea supuestos sobre esta medida y utiliza el recíproco de ésta como una medida de potencial presencia de ironía. Con lo anterior en mente, entre menor sea la similitud semántica, menor será la coherencia en el contexto y por lo tanto mayor será el desequilibrio en el contexto, por lo que existiendo oposiciones en el contexto, es probable que haya presencia de ironía. Como un ejemplo, a continuación se comparan los puntajes de similitud semántica de un texto potencialmente irónico con el de un texto objetivo, en los cuales se da el resultado esperado según los supuestos:

  - **Irónico:** Me encanta mi generosa y asquerosa alergia =)
    Puntaje de similitud semántica: 0.0235
    Puntaje Contextual Imbalance: 42.5532
  - **Objetivo:** Chile: destino 2015 imperdible según medios internacionales
    Puntaje de similitud semántica: 1.1359
    Puntaje Contextual Imbalance: 0.8804

  Como se aprecia en este caso, el texto irónico tiene un puntaje de similitud semántica mayor que el puntaje del texto objetivo, por lo tanto, tendrá un puntaje de desequilibrio contextual mayor, indicando que es más probable que sea irónico. Por supuesto, este comportamiento no es absoluto en todos los datos, sin embargo, es probable que se

presente de forma más frecuente en los textos irónicos según los supuestos del modelo, por lo que eso es lo que este atributo pretende capturar.

Temporal Imbalance: Este atributo pretende capturar la presencia de oposición, pero a diferencia del atributo anterior donde se identifica oposición a nivel del contexto, acá se busca identificar oposición a través del tiempo, específicamente comparando el pasado con el presente, como se plantea en [42]. Para esto, se analizan las polaridades de sentimientos en los verbos conjugados en pasado y presente que componen cada texto analizado para reflejar eventuales inconsistencias. Se dará un ejemplo detallado de este atributo en el Capítulo 5.

- **Style:** La dimensión *Style* busca determinar los elementos propios de la sintaxis utilizada al momento de expresar ironía, caracterizando secuencias de caracteres que sean utilizadas frecuentemente. Estos elementos incluyen cadenas de caracteres, cadenas de palabras y cadenas con una representación abstracta de las palabras en cada texto analizado, utilizando la polaridad de sentimientos que estas transmiten.

  Character N-grams: El primer atributo de esta dimensión se basa en la caracterización de n-gramas de caracteres (de forma abreviada *cgrams*). Estos corresponden a cadenas de n caracteres extraídos desde un texto. Estas estructuras han sido ámpliamente utilizadas en distintas aplicaciones de categorización de texto, como identificación de autores de textos [52], identificación de spam en correos electrónicos [53], entre otros. Su uso se relaciona con la efectividad que estas unidades tienen para capturar los detalles del estilo de un autor [54]. Al igual que en [5], se consideran cadenas entre 3 y 5 caracteres. Por ejemplo:

  – Que hermosa mañana
    3-cgram: que, ue_, e_h, _he, her, erm, rmo, mos, osa, sa_, a_m, _ma, mañ, aña, ñan, ana
    4-cgram: que_, ue_h, e_he, _her, herm, ermo, rmos, mosa, osa_, sa_m, a_ma, _mañ, maña, añan, ñana
    5-cgram: que_h, ue_he, e_her, _herm, hermo, ermos, rmosa, mosa_, osa_m, sa_ma, a_mañ, _maña, mañan, añana

  En los ejemplos se puede apreciar que se consideran los espacios entre palabras para la construcción de los n-gramas (reemplazándolos con un guión bajo), ya que es propio de este tipo de unidades incluirlos.

  Skipgrams: Este atributo se enfoca en capturar *skipgrams*. Los skipgrams corresponde a cadenas de las palabras que constituyen un texto, pero además incluyendo las cadenas producidas al omitir ciertas palabras (saltarlas). Formalmente un skipgram se define como [55]:

$$\left\{ w_{i1}, w_{i2}, \ldots, w_{in} | \sum_{j=1}^{n} i_j - i_{j-1} < k \right\} \tag{4.1}$$

  En la ecuación 4.1, k corresponde a la distancia del salto entre las palabras del texto w_i. En base a esto, se aprecia que un skipgram con saltos de largo $k > 0$ incluye además todos los skipgrams que se formarían con saltos menores a k. Para esta aplicación se consideran solo bigramas (o sea, cadenas de dos palabras) que consideren saltos de 3

palabras ya que secuencias más largas son poco comunes [5]. Esto incluirá aquellas cadenas formadas por saltos de dos palabras, una palabra y ninguna palabra (o sea, n-gramas de palabras). Por ejemplo:

- Esta mañana es perfecta en todo sentido
  3-skipbigrams: esta mañana, esta es, esta perfecta, esta en, mañana es, mañana perfecta, mañana en, mañana todo, es perfecta, es en, es todo, es sentido, perfecta en, perfecta todo, perfecta sentido, en todo, en sentido, todo sentido

Polarity Skipgrams: Por último, este atributo pretende caracterizar un tipo de estructura diferente a las anteriores. Utiliza el mismo concepto de skipgram utilizado anteriormente, pero lo aplica a una cadena formada en base a las etiquetas de polaridad de sentimientos de cada una de las palabras de un texto. Cada palabra es etiquetada como *positiva*, *negativa* y luego se extraen los skipgrams de las cadenas resultantes. Al igual que el atributo anterior, se consideran saltos de 3 palabras pero como es necesario etiquetar palabras según su polaridad de sentimientos, se realiza borrado de stopwords. Un ejemplo:

- Me está entrando fiebre, el dolor de garganta aumenta y mi cabeza va a explotar. Genial
  Palabras identificadas: dolor, aumentar, explotar, genial
  Etiquetas: negativo, positivo, negativo, positivo
  Polarity skipgrams: negativo positivo, negativo negativo, negativo positivo, positivo negativo, positivo positivo, negativo positivo

Es importante señalar que no se incluye una etiqueta para todas las palabras resultantes del borrado de stopwords. Esto se debe a la capacidad de las herramientas utilizadas para el etiquetado de sentimientos. En el Capítulo 5 se dan a conocer mayores detalles respecto a éstas.

- **Emotional Context:** Finalmente, esta dimensión trata de transmitir información relacionada con los estados emocionales de las personas. Existen elementos textuales que ya cumplen con estas funciones, tales como los emoticonos. Sin embargo, esta dimensión trata de ir más allá de elementos propios del lenguaje o de la polaridad de las palabras. Lo que se busca es caracterizar la expresión de ironía en función de abstracciones de las emociones expresadas con cada palabra, desde una perspectiva más cercana a la psicología. Para esto, se utiliza una versión en Español del *Dictionary of Affect in Language* el cual a distintas palabras añade un puntaje de 1 a 3 en función de tres aristas definidas en el trabajo original desarrollado por Whissell en [56], donde se caracterizan las emociones que perciben las personas con cada una al enfrentarse a una palabra en específico. Cada arista define uno de los atributos de esta dimensión, los cuales son descritos a continuación:

  Activation: Este atributo hace referencia al grado de respuesta (activo o pasivo) que tiene un humano en un estado emocional respecto a la palabra en cuestión.

  Imagery: Este atributo trata de reflejar qué tan fácil o difícil es para un humano realizar una imagen mental frente a una palabra dada.

  Pleasantness: Finalmente, este atributo cuantifica el grado de placer o agrado que representa una palabra para el humano.

Los tres atributos son representados de la misma forma, por un vector compuesto por los puntajes correspondientes asociados a cada palabra. Luego, se obtiene el promedio de los valores que componen este vector para caracterizar el atributo. Por ejemplo:

- Para conseguir un crédito debes demostrar al banco que no lo necesitas
  Vector Activation: 2.5000, 2.0000, 3.0000, 1.4000, 3.0000
  Vector Imagery: 2.6667, 2.8000, 1.8000, 2.6000, 2.0000
  Vector Pleasantness: 2.3333, 1.6000, 2.4000, 2.4000, 2.4000
  Esto da como resultado puntajes de 2.38, 2.37 y 2.23 para los atributos Activation, Imagery y Pleasantness respectivamente.

De forma similar a como podría pasar en los atributos Temporal Imbalance o Polarity Skipgrams, la asignación de puntajes a cada palabra se realiza dentro de las limitaciones de la herramienta utilizada para ello por lo que es esperable que no sea posible incluir todas las palabras (sin considerar stopwords) que constituyen un texto.

4.2 Método de Clasificación

A continuación se describe el método de clasificación utilizado para realizar la predicción de textos irónicos. El método corresponde a Naive Bayes el cual utiliza una distribución Multinomial para poder manejar variables que representan las frecuencias de los patrones, más allá de variables binarias como se utiliza comúnmente.

4.2.1 Clasificador Naive Bayes

Un clasificador Naive Bayes es basado en el Teorema de Bayes, como se explica en el Capítulo 2. Al aplicarlo en un problema de clasificación, a grandes rasgos lo que se pretende es descubrir la probabilidad de que se cumpla una *hipótesis* H en función de una *tupla* de datos X (medidas en un set de n atributos). Así, el valor buscado corresponde a $P(H \mid X)$ llamada *probabilidad a posteriori* de H condicionado a X. Este valor puede obtenerse a través de Bayes:

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)} \tag{4.2}$$

En base a la idea anterior, un clasificador Naive Bayes es desarrollado a través de 5 etapas [15]:

1. Primero se definen los datos con los que se trabajará. Sea D un set de entrenamiento, correspondiente a un conjunto de tuplas junto a su respectiva clase. Cada tupla corresponde a un vector de n atributos $X = (x_1, x_2, ..., x_n)$.

2. Consideremos que se tienen m clases $C_1...C_m$. Entonces dado X, el clasificador predice que X pertenece a la clase que tenga la más alta probabilidad a posteriori condicionada a X. Entonces se predice que X pertenece a la clase C_i si y solo si:

$$P(C_i \mid X) > P(C_j \mid X) \text{ para } 1 \leq j \leq m, j \neq i \tag{4.3}$$

   Para obtener lo que propone 4.3, se utiliza Bayes:

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)} \tag{4.4}$$

3. El objetivo es maximizar 4.4 utilizando la condición que propone 4.3. Se debe considerar que una tupla podría pertenecer a cualquier clase, por lo que $P(X)$ es constante para todos los casos. Por esto, para maximizar $P(C_i \mid X)$ solo es necesario maximizar la expresión $P(X \mid C_i)P(C_i)$. Si $P(C_i)$ no es conocida, en general son asumidas iguales para todas las clases por lo que el problema se reduce a maximizar $P(X \mid C_i)$.

4. Al tener sets con muchos atributos, el cálculo de $P(X \mid C_i)$ podría ser computacionalmente costoso. Para reducir ese costo, este clasificador asume como supuesto *ingenuo* una independencia condicional entre las clases, lo que se traduce en que los valores de atributos son condicionalmente independientes entre ellos, en función de una clase. Bajo este supuesto, $P(X \mid C_i)$ se puede calcular como:

$$P(X \mid C_i) = \prod_{k=1}^{n} P(X_k \mid C_i) \tag{4.5}$$

   El cálculo de los valores $P(X_k \mid C_i)$ son estimados fácilmente y según las características del clasificador. Para este caso, se pretende considerar atributos basados en frecuencias de patrones más que su simple presencia (variable binaria). Para esto, los valores $P(X_k \mid C_i)$ son calculados a través de una distribución de probabilidad Multinomial (una generalización de la distribución Binomial). Esta distribución es parametrizada por medio de $\theta_{C_i} = (\theta_{C_i 1}, \theta_{C_i 2}, ..., \theta_{C_i n})$ donde $\theta_{C_i k} = P(X_k \mid C_i)$. Para estimar $\theta_{C_i k}$ se utiliza un método suavizado de máxima verosimilitud:

$$\hat{\theta}_{C_i k} = \frac{N_{C_i k} + \alpha}{N_{C_i} + \alpha n} \tag{4.6}$$

   Donde $N_{C_i k}$ corresponde al número de veces que aparece el atributo k en la clase C_i y N_{C_i} corresponde a conteo total de todos los atributos en la clase C_i.

5. Por último, es necesario considerar que existe la posibilidad de que para algún atributo k no existan tuplas que lo posean en una clase C_i, por lo que para ese atributo $P(X_k \mid C_i) = 0$ y dada la definición en 4.5, esto provocaría que $P(X \mid C_i) = 0$. Para evitar esto, se realiza el *Suavizado de Laplace* y en este caso en particular, corresponde a definir $\alpha = 1$ para la expresión 4.6.

Capítulo 5

Implementación

Para poder manejar los sets de datos descritos anteriormente y realizar la extracción de los atributos que componen el modelo de detección de ironía en función de ellos, es necesario implementar un sistema para dar soporte a ese proceso, utilizando diversas herramientas tecnológicas. A continuación se describe el diseño e implementación de éste sistema, junto a la aplicación de las tecnologías necesarias para esto. Este capítulo dará una breve introducción a estas herramientas y además describirá la lógica que sigue el proceso de extracción y clasificación junto con el rol de estas herramientas en cada uno.

5.1 Herramientas Tecnológicas

A continuación se describirán a grandes rasgos las herramientas principales que fueron utilizadas para el desarrollo de los módulos que serán descritos en este capítulo.

- **Python**: Es un lenguaje de programación de propósitos generales, interpretado y orientado a objetos. Entre sus características es posible destacar su código diseñado para ser uniforme, mantener un orden y ser muy fácil de entender. En general, se requieren menos líneas de código para realizar tareas que requerirían una cantidad mucho más alta de código en otros lenguajes, por lo que también es bastante productivo a la hora de programar. Python incluye una librería estándar con múltiples funcionalidades y además tiene un alto soporte de desarrolladores de paquetes o librerías de diversa especie, generalmente de libre acceso lo que lo hace aun más atractivo [57]. En esta oportunidad es utilizado dado que facilita el desarrollo, es rápido de aprender y además las librerías que lo complementan son fundamentales para esta implementación.
- **JSON**: JavaScript Object Notation es un formato de transferencia de datos, muy ligero y fácil de usar. Es independiente a cualquier lenguaje de programación, pero sus convenciones facilitan su uso con cualquier tipo de lenguaje.[1] La API de Twitter entrega los datos en este formato, por lo que es necesario utilizarlo.
- **TreeTagger**: Es una herramienta que permite realizar anotaciones en un texto respecto

[1] http://json.org/

a los POS-Tags y los lemas de las palabras que le componen. Fue desarrollada por Helmut Schmid en el Institute for Computational Linguistics de la Universidad de Stuttgart[2]. Se encuentra disponible en varios idiomas incluyendo el Español. Para su uso, se utilizó un *wrapper* que adapta sus funcionalidad para poder ser utilizado en Python[3], ya que originalmente TreeTagger funciona en el lenguaje Perl.

- **MySQL**: Corresponde a un sistema de administración de bases de datos relacionales, altamente popular y simple de usar. Es rápido y puede manejar grandes cantidades de información sin requerir un hardware muy sofisticado, su desarrollo está en sincronía con los estándares y su interfaz es versátil a la hora de trabajar con distintos lenguajes de programación [58]. En particular, para su uso se utiliza el paquete de servidores XAMPP[4] y la librería MySQLdb[5] de Python.
- **Java**: Java es un lenguaje orientado a objetos, de alto nivel (más cercano a los usuarios que al comportamiento de la máquina) pero a diferencia de Python, es un lenguaje compilado. Los programas escritos en Java son compilados y llevados desde el código de alto nivel, a un programa más cercano al lenguaje de una máquina. Luego son interpretados para poder ser utilizados por una máquina virtual propia. Java puede ser utilizado en cualquier computador que disponga de este interprete, lo cual es atractivo para el desarrollador [59]. En esta implementación, Java es utilizado de forma muy breve.
- **NLTK**: Natural Language Toolkit es una librería de Python que incluye una gran cantidad de funcionalidades que permiten a Python trabajar con recursos lingüisticos [6]. En este caso, NLTK es principalmente utilizado para tareas de pre-procesamiento de texto.
- **scikit-learn**: Es una librería especializada en funciones de Machine Learning para Python, la cual provee diversos algoritmos para tareas de clasificación, regresión y clustering así como funciones pensadas en el procesamiento de datos, extracción de atributos e incluso la evaluación de los procesos de clasificación [60]. La clasificación y su evaluación es implementada a través de esta librería.
- **NumPy**: Es una librería de Python para realizar computación científica. Sus funciones permiten trabajar de forma natural con arreglos y matrices de gran tamaño [61]. NumPy es utilizado por scikit-learn para trabajar, por lo que su uso es requerido en esta implementación.

5.2 Organización y Evaluación de Tweets

Una vez que se han extraídos los tweets utilizando los crawlers y los recursos descritos en el Capítulo 3, es necesario organizar estos datos para su utilización. Para esto, el proceso de divide en tres pasos:

[2]http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/
[3]https://perso.limsi.fr/pointal/dev:treetaggerwrapper
[4]https://www.apachefriends.org/es/index.html
[5]http://mysql-python.sourceforge.net/MySQLdb.html
[6]http://www.nltk.org/

1. El primer paso consiste en almacenar la información que se utilizará de cada tweet recolectados con los crawlers. Inicialmente los tweets son almacenados en formato JSON, utilizando archivos de texto plano. Posteriormente atraviesan un proceso de filtrado como se describe en el Capítulo 3 donde los tweets seleccionados son almacenados en nuevos archivos de texto plano conservando su formato JSON. Finalmente, mediante un script en Python se extraen los campos *id* y *text* correspondientes a la ID de Twitter que cada tweet posee y el texto que compone al tweet. La ID de Twitter se conserva para poder acceder a todos los datos de un tweet en particular, en caso de ser necesario en el futuro. Además, este script inserta los datos en una base de datos MySQL, específicamente en la tabla *tweet* descrita por la Figura 5.1. Para ello utiliza la siguiente consulta SQL:

   ```
   INSERT INTO tweet (twitter_id, text, ev_1, ev_2, ev_final, label)
   VALUES (var1, var2, var3, var4, var5, var6)
   ```

   Donde $var1$, $var2$, $var3$, $var4$, $var5$ y $var6$ corresponden a variables en Python. En este caso solo se completan los campos *twitter_id* y *text* por lo que las variables $var3$, $var4$, $var5$ y $var6$ se mantienen en blanco. Cabe mencionar que el campo *id* que se aprecia en la Figura 5.1 no se completa en esta consulta ya que tiene la característica $AUTO\_INCREMENT$ por lo que su valor parte desde el número entero 0 y aumenta automáticamente a medida que se añaden nuevos registros a la tabla.

2. El siguiente paso es añadir los valores de las evaluaciones descritas en la parte final el Capítulo 3. Los resultados de estas evaluaciones fueron almacenadas en un archivo CSV (separado por comas) y posteriormente leídas mediante un script en Python para extraer su información. La estructura del archivo tiene 3 columnas, en las que las dos primeras columnas contienen los resultados de la primera evaluación, mientras que la tercera columna tiene los resultados de la segunda evaluación. Como se describe en el Capítulo 3, no todos los tweets pasan por la segunda evaluación, por lo que esta columna podría contener valores vacíos. Luego, se realiza una iteración a través de todas las filas obtenidas desde el archivo CSV para insertar estos resultados en la tabla *tweet* por medio de una consulta SQL:

   ```
   UPDATE tweet SET ev_1 = var1, ev_2 = var2, ev_final = var3, label = var4
   WHERE id = var5
   ```

   Las variables $var1$, $var2$ y $var3$ corresponden a los valores del resultado de la evaluación, extraídos desde el archivo CSV. Para mantener un orden similar a éste, las variables $var1$ y $var2$ tienen los resultados de la primera evaluación, y la variable $var3$ tiene el resultado de la segunda evaluación. $var4$ es una variable binaria (0 o 1) y esta varía dependiendo de los resultados de la evaluación. Si $var1 = var2$, $var4$ toma un valor dependiendo de $var1$ (o $var2$, es indiferente) siendo 1 si $var1 = Y$ o 0 si $var1 = N$. De otro modo, si $var1 \neq var2$ ahora $var4$ dependerá de $var3$ (el contenedor de la segunda evaluación) y de una forma similar al caso anterior, $var4$ será 1 si $var3 = Y$ o 0 si $var3 = N$. Cabe mencionar que en todos los casos en que $var1 \neq var2$, $var3$ es distinto de vacío.

3. Por último, como paso final es necesario añadir aquellos tweets que forman parte del set negativo construido en base a medios de prensa. El procedimiento es idéntico al primer paso, con una diferencia en las variables de su consulta SQL:

```
INSERT INTO tweet (twitter_id, text, ev_1, ev_2, ev_final, label)
VALUES (var1, var2, var3, var4, var5, 0)
```

En esta oportunidad, solo $var3$, $var4$ y $var5$ se mantienen en blanco mientras que $var6$ ahora toma valor igual a 0 ya que todos los tweets pertenecientes a este set corresponden a casos negativos o no irónicos.

El procedimiento descrito se realiza en orden por lo que los tweets correspondientes a los sets positivos y negativos obtenidos mediante la evaluación con seres humanos se encuentran entre las ID 1 y 17277 de la tabla *tweet* mientras que desde la ID 17278 hasta la ID 35108 corresponde al set de datos negativo extraído desde medios de prensa. El proceso descrito anteriormente se esquematiza en la Figura 5.3.

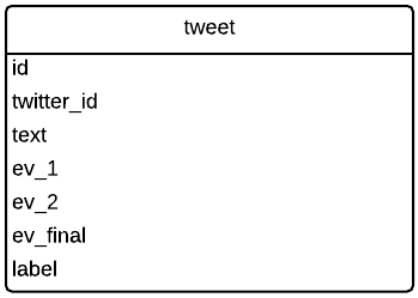

Figura 5.1: Tabla *tweet* donde se almacenan datos asociados a los tweets recolectados para la clasificación
Fuente: Elaboración Propia.

5.3 Módulo de Extracción de Atributos

La implementación más exhaustiva de este trabajo corresponde al módulo de extracción de atributos. Como se explica en el Capítulo 4, el modelo consta de distintos atributos y cada uno tiene sus propias características y dificultades asociadas a su implementación. Para el desarrollo de este módulo se utilizó en mayor parte scripts de Python donde se creó una función para extraer y caracterizar cada uno de los atributos como el valor que debe tener en el vector de atributos que se utilizará en la clasificación. Cada función retorna su vector de atributos correspondiente al término de su extracción. A continuación se explicará en detalle el proceso que cada una de estas funciones sigue, junto con la explicación de cómo se implementó.

5.3.1 Modelo de Datos

Antes de pasar a la explicación de la extracción de cada atributo, es importante describir el modelo de datos que sustenta este módulo. Mediante las funciones desarrolladas, los atributos son extraídos y luego un script en Python junta esta información y la almacena en este modelo de datos para que la información se preserve luego del proceso y pueda ser utilizada en el módulo de clasificación.

El modelo cuenta con la tabla *tweet* utilizada para el almacenamiento de la información relevante para cada tweet y además cuenta con una tabla para cada uno de los 11 atributos que componen el modelo. Cada tabla contiene al menos 3 campos como se aprecia en la Figura 5.2:

- *id*: Corresponde a la ID única para cada campo (llave primaria). Es de tipo INT (número entero).
- $feature\_vector$: Corresponde al vector de atributos. Cada vector de atributos tendrá una forma distinta, dependiendo de las necesidades del atributo en cuestión. Es de tipo LONGTEXT, ya que algunos vectores son demasiado grandes.
- *tweet_id*: Corresponde a una llave foránea que vincula la tabla de atributos correspondiente con la tabla *tweet*. Como hace referencia a una ID, su tipo es INT.

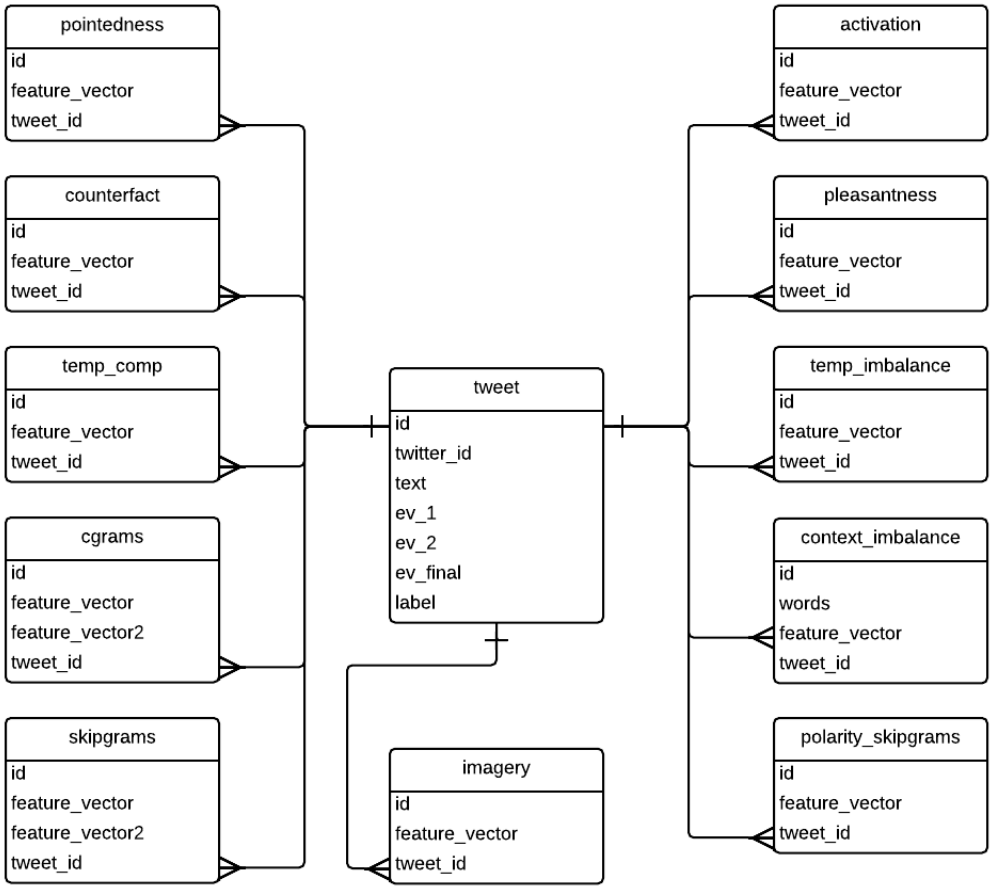

Figura 5.2: Modelo de datos utilizado en la extracción de atributos
Fuente: Elaboración Propia.

Cada tabla tiene una relación $1 - N$ con la tabla *tweet*. Existen dos tablas que tienen más de 3 campos, correspondientes a las tablas *context_imbalance*, *cgrams* y *skipgrams*. La primera requiere el campo extra *words* ya que tiene un tratamiento especial para la obtención de su vector de atributos. Las dos últimas requieren un vector de atributos extra debido a las

pruebas que se realizarán. Las razones de esto se explican en el detalle de la implementación de la extracción de cada atributo.

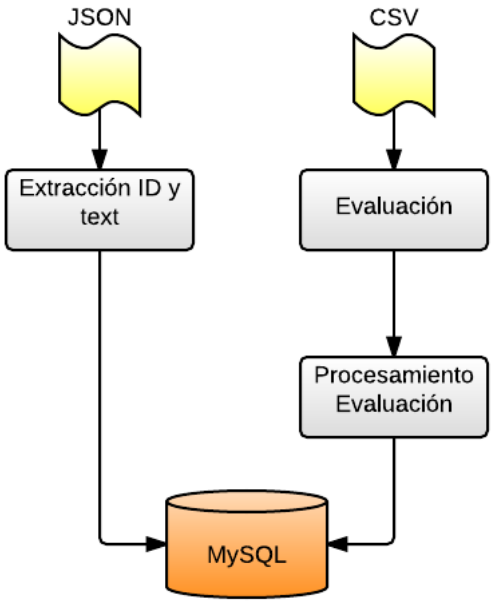

Figura 5.3: Esquema de Organización y Evaluación de Tweets
Fuente: Elaboración Propia.

5.3.2 Pre-procesamiento Básico

Para la extracción de atributos, los textos utilizados deben pasar por un pre-procesamiento básico para poder obtener la información que requiere cada atributo. Este proceso se utiliza de forma recurrente en las funciones de extracción por lo que se explicarán previamente en detalle.

1. **Tokenización**: El primer paso corresponde a una *tokenización*, la cual se realiza utilizando el paquete *NLTK* de Python. Específicamente se utiliza la función $regexp\_tokenize()$ [29] que permite realizar una tokenización en base a una *expresión regular*. Una expresión regular es una secuencia de caracteres que conforma un patrón de búsqueda, el cual permite identificar patrones presentes en otras secuencias de caracteres o *strings*. La expresión regular utilizada permite separar un texto en sus palabras, en caracteres no alfabéticos como números y signos de puntuación, y además permite caracterizar *emoticonos*. Esta expresión regular es incluida en el Apéndice A .
2. **Identificación y Filtrado de Emoticonos**: Un emoticono es un ícono emocional, el cual puede ser utilizado por un usuario de Twitter (o de cualquier otro medio) para transmitir sus emociones. En la tokenización se separan estos emoticonos y luego para filtrarlos se utiliza una lista de emoticonos construida en base a la lista propuesta por [5]. Sin embargo, esta lista es complementada con otros tipos. Los emoticonos incluidos se pueden diferenciar en 4 categorías: Emoticonos hacia la Izquierda, Emoticonos hacia la Derecha, Emoticonos Frontales y Otros. La lista completa se incluye en el Apéndice A .

3. **Lematización y POS-Tagging**: Durante este pre-procesamiento es muy relevante llevar las palabras a su raíz gramatical para poder trabajar con ellas, lo que puede obtenerse mediante una lematización (llamaremos *lema* a esta raíz). Además en algunas funciones se requiere del POS-Tag asociado a cada palabra. Ambas cosas pueden obtenerse utilizando la herramienta TreeTagger. El output de TreeTagger entrega la palabra original, junto a el POS-Tag de la palabra y su lema. En caso de que la palabra no sea reconocida, TreeTagger no entregará el POS-Tag y como lema entregará *<unknown>*. Esto suele pasar con palabras desconocidas para la herramienta, como también con palabras incompletas o con faltas de ortografía.
4. **Eliminación de Caracteres No Alfabéticos**: Luego de la tokenización, existen *tokens* que no son útiles para la extracción de atributos (o al menos, para algunas funciones no lo son) y estos corresponden a cadenas que se componen exclusivamente de caracteres *no alfabéticos* como números, signos de puntuación y otros caracteres especiales. Para su eliminación, se recurre a las funciones *is_alpha*() y *is_digit*() proveídas por Python, las cuales permiten identificar caracteres alfabéticos y dígitos respectivamente.
5. **Eliminación de Palabras Desconocidas**: Muchas de las funciones utilizadas para la extracción de atributos realizan una identificación de palabras conocidas en base a otra lista, para añadirles ciertas características. Por esta razón, las palabras que tienen un lema desconocido no sirven para el proceso y son eliminadas. Por eso, todas aquellas palabras que tengan un lema igual a *<unknown>* son descartadas.

A pesar de que este proceso es descrito en orden, no todos los pasos son seguidos en todas las funciones. Los pasos son seguidos según la necesidades de pre-procesamiento de cada función de extracción. El uso de estos pasos será mencionado en el detalle para cada función del módulo de extracción, descrito a continuación.

5.3.3 Signatures

A continuación se explica la extracción de atributos de la dimensión *Signatures*. Recordemos que los atributos correspondientes a esta dimensión son *Pointedness*, *Counterfactual* y *Temporal Compression*. El vector de atributos para esta dimensión es construido en función de frecuencias, es decir, qué tantas veces está presente alguna de las características que se pretende capturar en esta dimensión.

1. **Pointedness**: Este atributo se preocupa de detectar uso de emoticonos, uso de signos de puntuación y comillas, y el uso de palabras escritas en mayúsculas. Para realizar esta detección, basta usar los pasos descritos en el pre-procesamiento junto con la función *is_upper*() de Python, la cual identifica cuando un string se compone de caracteres escritos en mayúsculas. Para construir el vector de atributos, se utiliza un vector de $1x3$ en la que cada valor corresponde a un número entero mayor o igual a cero, la cual representa la frecuencia con la que aparece un emoticono, un signo de puntuación o comillas, y una palabra en mayúsculas respectivamente. Entonces, viendo el ejemplo que aparece en el Capítulo 4:
*Mi autobus retrasado**???** **NOOO** me estan mintiendo verdad**??** Pero si eso **JAMAS** sucede ¬¬ **!!!***

Para este caso, el vector de atributos será $[1, 3, 2]$. Estos valores son concatenados utilizando comas mediante la función $join()$ de Python, para posteriormente retornar el string resultante.

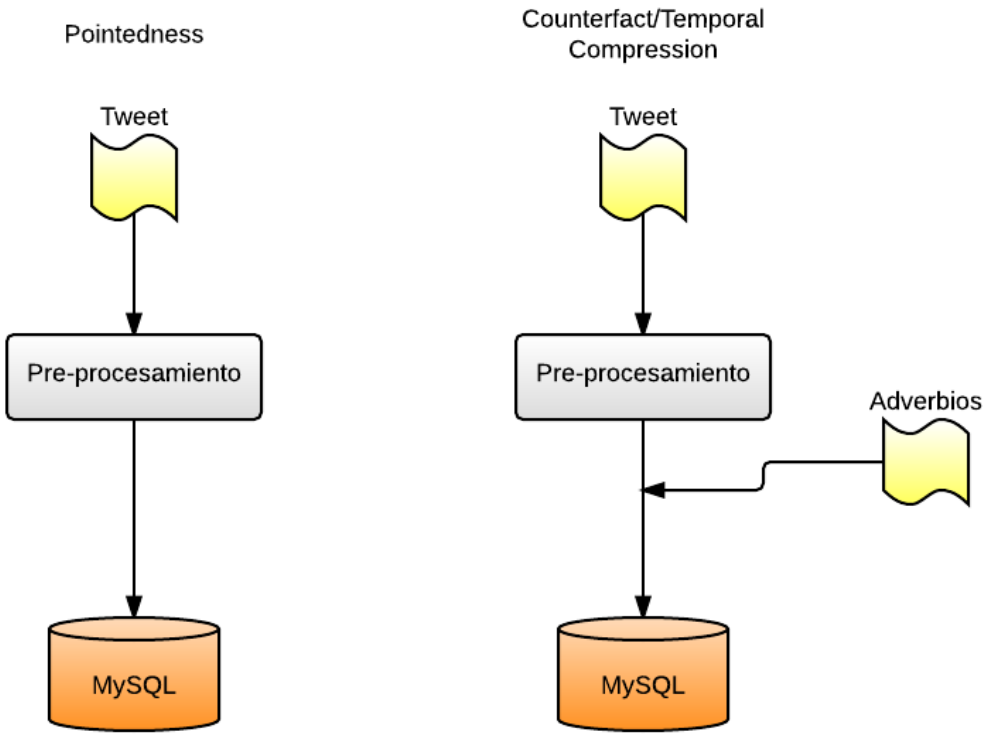

Figura 5.4: Esquema dimensión Signatures
Fuente: Elaboración Propia.

2. **Counterfactual y Temporal Compression**: En la función de este atributo, se realiza el pre-procesamiento completo para cada texto. De esta forma se tienen las palabras válidas junto a su POS-Tag. Este último es muy importante en este atributo ya que en este caso se está buscando adverbios. La lista de adverbios utilizada para buscar coincidencias corresponde a una adaptación al idioma Español de la lista propuesta en [5]. Para cada palabra se respetó su rol gramatical, considerando que las palabras incluidas fueran adverbios. La lista completa se encuentra en el Apéndice A .
Para realizar la identificación, la lista de adverbios fue dividida en tres tipos, según la cantidad de palabras que componen cada adverbio y así, en esta lista existen adverbios de tres, dos y una palabra. Es importante notar que existen adverbios de más de una palabra en los que una de sus palabras corresponde a un adverbio por si misma (por ejemplo, el adverbio de tres palabras *más o menos* tiene las palabras *más* y *menos*, que también son adverbios) por lo que para evitar que la misma palabra fuese contada más de una vez, a medida que los adverbios son identificados en un string, estos van siendo retirados de éste. La identificación comienza con los adverbios en los que se da este caso, para luego seguir con los demás en orden decreciente según la cantidad de palabras que los componen.

Cabe mencionar que para disminuir el impacto de la presencia de faltas de ortografía asociadas al uso de tildes, en las listas de identificación se incluyeron aquellos adverbios

que podrían ser escritos sin incluir su tilde. Por último, para la identificación de los adverbios de una sola palabra, además se verifica que el POS-Tag asignado a la palabra por TreeTagger, correspondiera a *ADV* que es el POS-Tag que TreeTagger asigna a los adverbios en Español. El vector de atributos en esta oportunidad corresponde a un número entero mayor a cero, que representa la cantidad de adverbios identificados de la lista. Así, para uno de los ejemplos del Capítulo 4:

Los profesores nos llaman desactualizados porque no usamos como debemos las computadoras, ***sin embargo****,* ***solo*** *mandan trabajos a mano*

El vector de atributos será [2]. La función del atributo *Temporal Compression* es exactamente igual a la de *Counterfactual*, con la diferencia de que la lista de adverbios utilizada es distinta. Ésta también es incluida en el Apéndice A .

5.3.4 Unexpectedness

La siguiente dimensión en esta implementación corresponde a *Unexpectedness*, la cual se compone de los atributos *Contextual Imbalance* y *Temporal Imbalance*. A continuación se describen sus procedimientos.

1. **Contextual Imbalance**: La implementación de este atributo es el único que tiene dos partes utilizando lenguajes diferentes, lo cual fue necesario debido a las herramientas que se necesitaron para su desarrollo. La función en Python solo realiza el pre-procesamiento básico para cada texto y obtiene todas los lemas de las palabras que lo componen. Estas palabras son concatenadas en un string mediante la función *join*() y un separador, para el cual se utilizó el string <*SEP*>. El ejemplo mostrado en el Capítulo 4 quedaría de la siguiente forma:

   Original: Me encanta mi generosa y asquerosa alergia =)

   Resultado: yo<SEP>encantar<SEP>generoso<SEP>y<SEP>asqueroso<SEP>alergia

   El string resultante es insertado mediante la siguiente consulta SQL:

   ```
   INSERT INTO context_imbalance (words, tweet_id) VALUES (var1, var2)
   ```

   Donde *var*1 corresponde al string resultante y *var*2 corresponde a la ID del tweet original. El siguiente paso corresponde a tomar esta lista de palabras y encontrar los puntajes de similitud semántica entre éstas. Para realizar este paso, se utilizó una implementación propia de la herramienta *Spanish JavaSimLib*, descrita en el trabajo de Lozano et al. (2012) [62]. Esta es una utilidad desarrollada en Java, en base a otra utilidad existente en idioma Inglés, la cual utiliza la taxonomía de WordNet[7] para realizar el cálculo de una medida de similitud semántica entre dos palabras. Lozano utiliza una versión de Wordnet 1.6 en Español desarrollada por el *Center for Language and Speech Technologies and Applications (TALP)* de la Universidad Politécnica de Cataluña[8]

[7] http://wordnet.princeton.edu/
[8] http://www.talp.upc.edu/

y junto a esto utiliza Apache Lucene[9] 1.4.3, herramienta que permite generar índices para un conjunto de datos, con el objetivo de agilizar la búsqueda y disposición de éstos.

La implementación de Spanish JavaSimLib utilizada cuenta con exactamente las mismas herramientas que describe y utiliza Lozano en su trabajo. El output de la herramienta es el cálculo de la medida de similitud semántica Jiang-Conrath [63], la cual entrega un número positivo $x \in [0, \infty)$. Entre mayor es este número, mayor será la similitud semántica entre ambas palabras por lo que para este caso es relevante el recíproco de ese valor, según los supuestos del atributo descritos en el Capítulo 4.

Para determinar el vector de este atributo, sería útil definir una brecha entre los valores del recíproco de la medida Jiang-Conrath para saber cuales cumplen la condición y cuales no. Para determinar esto, se ajustan los valores entregados por la medida para que sus recíprocos se encuentren entre 0 y 1 tratando los valores con una *función exponencial*, la cual asegura que todos los valores sean mayores a 1 (esto fue necesario ya que la medida Jiang-Conrath puede entregar valores menores a 1) y como es una función estrictamente creciente para números mayores a cero, se asegura de que los puntajes mantengan su orden original. De esta forma, todos los recíprocos de estos valores se encontrarán entre 0 y 1. Luego de este ajuste, es posible determinar el vector de atributos usando 3 niveles: $[min, med, max]$. En primera instancia, los tres valores corresponden a 0 pero si el valor del atributo se encuentra entre cierto rango, uno de los valores será igual a 1. Los rangos son los siguientes:

Si $valor \leq \frac{1}{3}$ y $valor > 0$ entonces $min = 1$

Si $valor \leq \frac{2}{3}$ y $valor > \frac{1}{3}$ entonces $med = 1$

Si $valor \leq 1$ y $valor > \frac{2}{3}$ entonces $max = 1$

Entonces para realizar todo lo descrito anteriormente, se utiliza un script en Java que primero selecciona el campo *words* de cada tweet en la tabla *context_imbalance* mediante la consulta SQL:

```
SELECT * FROM context_imbalance ORDER BY id ASC
```

Luego, cada string obtenido es separado utilizando la función *split*() de Java, según el string <*SEP*>. Posteriormente, se eliminan todas aquellas palabras que no estén disponibles en la versión de WordNet utilizada por Spanish JavaSimLib ya que sería imposible utilizarlas para obtener algún valor. Posteriormente, se obtienen todos las combinaciones posibles (pares) entre las palabras disponibles junto a su cantidad, para luego calcular para cada par la medida Jiang-Conrath. De todos los valores obtenidos se obtiene el promedio, y luego la exponencial de ese promedio. Finalmente se obtiene el recíproco de este valor y se verifica su valor respecto a los rangos especificados para determinar el valor final del vector de atributos. Este valor final es insertado en la base de datos mediante la siguiente consulta SQL:

```
UPDATE context_imbalance SET feature_vector = var1 WHERE id = var2
```

[9]http://lucene.apache.org/

Con *var*1 igual al valor del vector de atributos concatenado por medio de la función *join*() y *var*2 igual a la ID del tweet en cuestión.

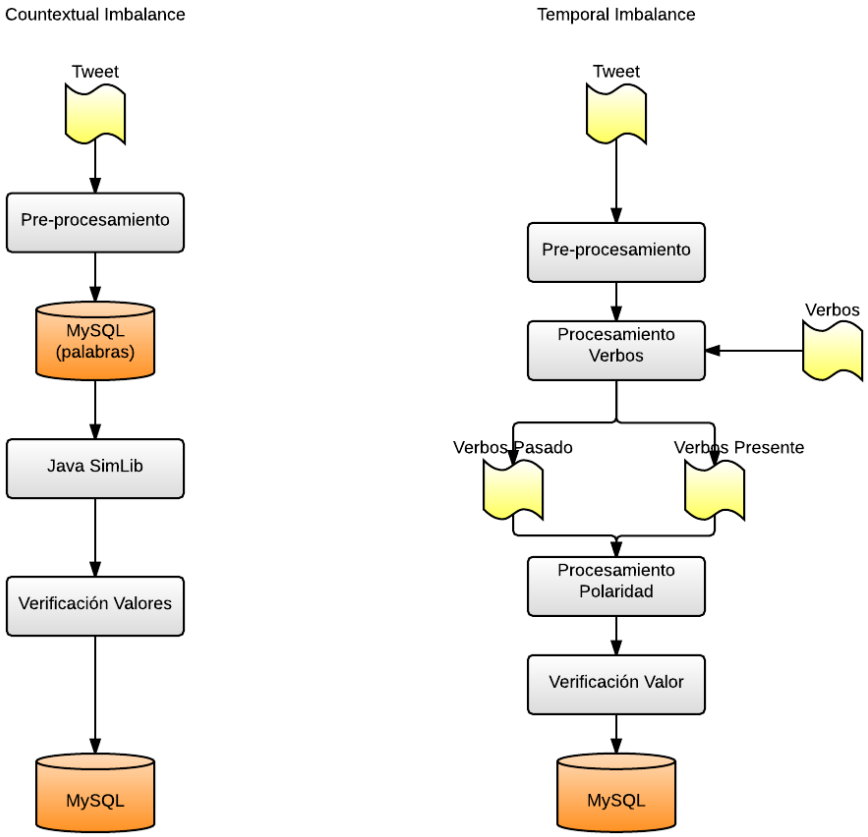

Figura 5.5: Esquema dimensión Unexpectedness
Fuente: Elaboración Propia.

2. **Temporal Imbalance**: Para determinar el vector de este atributo se realiza una identificación de todos los verbos presentes en el texto analizado. Luego, se determina cuáles de ellos se encuentran en tiempo pasado o presente, para ser consistentes con el criterio estipulado en el Capítulo 4 y, después, cada uno de éstos son etiquetados según su polaridad (POS y NEG para verbos positivos o negativos respectivamente). El valor del atributo es binario, al igual que el atributo *Contextual Imbalance* pero es determinado de forma distinta. Luego del etiquetado de polaridad de los verbos en pasado y presente, estos se analizan para identificar cualquier desequilibrio en las polaridades utilizadas en pasado, en comparación con las usadas en presente. En este caso, un desequilibrio es caracterizado por diferencias en las polaridades utilizadas en un tipo de verbos y otros. Por ejemplo, a continuación se ve un caso en el que existe un desequilibrio y otro caso en que no existe:

   EJ1: *ayer te* ***amaba*** *y hoy te* ***odio*** *:(*
   Verbos en pasado: amar (POS)
   Verbos en presente: odiar (NEG)

   EJ2: *Movilh* ***afirma*** *que evangélicos de Confamilia* ***respaldaron*** *polémicos dichos de Javier Soto*

Verbos en pasado: respaldar (POS)
Verbos en presente afirmar (POS)

Habrá un potencial desequilibrio cuando en el texto existan tanto verbos en tiempo pasado como en tiempo presente, pero si en el texto no está presente alguno de los tiempos, se asume que no es posible que exista un desequilibrio. El desequilibrio entonces ocurre cuando en alguno de los tiempos (pasado o presente) está presente un tipo de polaridad que no está presente en el otro tiempo, como se aprecia en los ejemplos. Si hay un desequilibrio, el vector de atributos corresponderá a [1] y en caso contrario será [0].

Para realizar la detección de estos desequilibrios el procedimiento es el siguiente. Primero se realiza el pre-procesamiento básico y se obtienen los lemas de todas las palabras del texto junto a su POS-Tag por medio de TreeTagger. Luego, se verifican cuales de estas palabras son verbos analizando el POS-Tag de cada palabra, donde si el primer carácter del tag corresponde a *"V"*, entonces la palabra corresponde a un tipo de verbo.

Luego, se deben identificar aquellos verbos conjugados en tiempo pasado o presente. Para esto, se utilizó un *lemario* (diccionario de lemas) de verbos en Español construido por Ismael Olea[10] como parte del proyecto *Recursos Lingüisticos Abiertos*[11] el cual contiene una lista de 10.783 verbos distintos en Español. Luego, en base a este se confeccionó una lista de conjugaciones en pasado y presente para todos los verbos encontrados de aquella lista, las cuales fueron construidas utilizando datos extraídos desde la página web de *Reverso*[12], por medio de un crawler web. Así, se recopilaron 204.193 conjugaciones para los verbos en pasado y 106.844 verbos en presente. Teniendo estos recursos disponibles, se generan dos listas con los lemas de los verbos que originalmente estaban conjugados en tiempo pasado o presente en el texto.

Finalmente, para realizar el etiquetado de polaridades se utilizaron dos recursos. El primero de ellos corresponde a *ML-Senticon*, un *Lexicon* de palabras en distintos idiomas (incluido el Español) a las que se le han asignado puntajes respecto a la polaridad percibida en ellos. Este fue construido en el trabajo de Cruz et al. (2014) en [64, 65] donde utiliza una adaptación del método utilizado para construir SentiWordNet[13] con propuestas de mejoras aplicadas por el autor. Los puntajes son distribuídos entre -1 y 1, correspondiendo a polaridad negativa cuando el puntaje es negativo, y polaridad positiva cuando el puntaje es positivo. ML-Senticon contiene 1.377 verbos con su respectiva evaluación. El segundo recurso corresponde a un Diccionario de Orientación de Sentimientos (*SO Dictionaries*) desarrollado por Brooke et al. (2009) [66] y proveídos por Maite Taboada, quien forma parte de la investigación. Este diccionario contiene 740 verbos con puntajes entre -5 y 5, donde el criterio para determinar la polaridad es el mismo que para ML-Senticon.

[10]https://github.com/olea/lemarios/
[11]https://forja.rediris.es/projects/rla-es/
[12]http://conjugador.reverso.net/conjugacion-espanol.html
[13]http://sentiwordnet.isti.cnr.it/

Los recursos fueron utilizados en conjunto. En el caso de los verbos, existen 191 palabras que están presentes en ambos textos pero solo 10 de esas palabras tienen discrepancias en su evaluación respecto a la polaridad de la palabra, lo que se considera un nivel de error aceptable. La función toma como prioridad a ML-Senticon para determinar la polaridad de una palabra y si esta no es encontrada en esta lista, la búsqueda se realiza en SO Dictionaries. Cabe mencionar que tanto la lista de conjugaciones como las listas de palabras junto a sus polaridades son recursos extremadamente escasos para el idioma Español, por lo que es de esperar que su desarrollo no sea tan completo como para otros idiomas, lo que puede provocar que no todas las palabras sean posibles de etiquetar por lo que en estos casos las palabras no son consideradas en el etiquetado final. Teniendo este recurso final y las listas de los lemas de verbos en pasado y presente, se determina el valor final del atributo y la función retorna este valor.

5.3.5 Style

La tercera dimensión del modelo corresponde a *Style* y es compuesta por tres atributos: *Character N-grams*, *Skipgrams* y *Polarity Skipgrams*. Sus funciones de extracción son descritas a continuación.

1. **Character N-grams**: Para la extracción de este atributo se utiliza el módulo *ngrams* del paquete NLTK, el cual puede obtener los N-gramas posibles desde una lista de datos. Como se busca obtener N-gramas de caracteres (de ahora en adelante serán llamados *cgrams*), las unidades que componen las listas de datos son los caracteres que componen el texto. A diferencia de procedimientos anteriores, no se realiza un preprocesamiento al texto para poder identificar patrones que combinen el uso de letras y signos gramaticales de otro tipo. El único tratamiento que se realiza al texto es llevar todas sus letras a minúsculas utilizando la función *lower*() de Python.

   Para caracterizar el vector correspondiente a este atributo, es necesario realizar un paso previo (para el atributo *Skipgrams* se realiza el mismo procedimiento). El vector de atributos se compondrá de la frecuencia de los cgrams detectados en el texto, a lo largo de un vector de números enteros. Este vector representa la presencia en el texto de cada uno de los cgrams relevantes detectados a lo largo del set de entrenamiento. Por este motivo, el primer paso para poder obtener estos vectores, es detectar todos los cgrams posibles de generar en el set de entrenamiento, junto con su frecuencia a lo largo de este set para luego generar un *vocabulario* de cgrams y conformar el vector para este atributo. Este tipo de vector de atributos es conocido como una representación *Bag-of-Words* de cada documento o texto, pero para que quede claro el procedimiento se utilizará un ejemplo sencillo en base a palabras. A continuación se asume que se tienen dos sentencias distintas que componen un set de entrenamiento:

   S1: *Cada país tiene su propia ley.*
   S2: *Cada pueblo tiene su propia cultura.*

   Luego, todas las palabras identificables entre ambas sentencias componen un vocabulario que se compone de las palabras junto a su frecuencia en el set de entrenamiento.

Se consideran los lemas de cada palabra para verificar su frecuencia:
`V = {cada : 2, tener : 2, su : 2, propia : 2,`
`país : 1, pueblo : 1, ley : 1, cultura : 1}`
Luego, en base a este vocabulario se construye un vector de dimensión $d = \|V\|$, con el que se puede realizar una representación de *Bag-of-Words* de cada documento:

	cada	tener	su	propia	país	pueblo	ley	cultura
S1	1	1	1	1	1	0	1	0
S2	1	1	1	1	0	1	0	1

Tabla 5.1: Ejemplo de representación Bag-of-Words para un conjunto de documentos.
Fuente: Elaboración Propia

El procedimiento para este atributo es análogo, con la diferencia de que la unidad son los cgrams y no las palabras. Luego de obtener el vector para un texto, los valores que lo componen son concatenados con comas utilizando la función $join()$ de Python, para que finalmente la función retorne el string resultante. Cabe mencionar que el vector de atributos depende del set de entrenamiento utilizado y como en este estudio se contemplan dos pruebas distintas, la tabla *cgrams* contiene dos campos para almacenar los vectores de atributos asociados a cada prueba. Lo mismo pasa con el atributo *Skipgrams*.

2. **Skipgrams**: Para la obtención de este atributo se realiza un procedimiento muy similar al atributo anterior, con la diferencia de que la unidad para conformar las representaciones Bag-of-Words corresponde a un *skipgram*, tal como se define en el Capítulo 4. Esta vez los textos pasan por el proceso de pre-procesamiento ya que solo son de interés las palabras para la construcción de los skipgrams. Entonces se realiza una tokenización y se eliminan los emoticonos, los tokens que se componen solo de números o caracteres no alfabéticos y luego se transforman todos los caracteres de todos los tokens a minúsculas utilizando la función $lower()$. En esta oportunidad no se utiliza TreeTagger.

   Para generar los skipgrams simplemente se itera sobre la lista de palabras resultante del pre-procesamiento, construyendo bi-gramas de palabras y realizando hasta 3 saltos a partir de cada palabra (esto incluiría los skipgrams de dos saltos). La iteración continúa hasta que ya no hay skipgrams posibles de generar. El resultado es ejemplificado en el Capítulo 4.

3. **Polarity Skipgrams**: El procedimiento para este atributo es algo distinto a los dos anteriores. Ahora se buscan skipgrams basados en etiquetas de polaridad y como se explica en el Capítulo 4, los skipgrams siempre son bigramas (una cadena de dos palabras o unidades). Ante esto, la representación Bag-of-Words para cada texto siempre se compondrá en base al siguiente vector:

   `V = {POS POS, POS NEG, NEG POS, NEG NEG}`
   Para este atributo, los textos son pre-procesados con el objetivo de obtener una lista con los lemas de todas las palabras que lo componen junto a su respectivo POS-Tag, por lo que se realiza una tokenización, para luego eliminar emoticonos y tokens no alfabéticos y finalmente obtener los lemas utilizando TreeTagger. Las palabras desconocidas y los signos de puntuación también son eliminados. Una vez que se tiene esta lista, es

posible realizar el etiquetado de polaridad utilizando los mismos recursos que en el atributo *Temporal Imbalance* (ML-Senticon y SO Dictionaries) pero esta vez no solo se consideran las listas de verbos, sino que también las listas de sustantivos, adjetivos y adverbios. El criterio para etiquetar es el mismo utilizado en ese caso, donde primero se considera ML-Senticon para obtener el puntaje y si la palabra no es encontrada en este recurso, se realiza la búsqueda en SO Dictionaries.

	Cant. ML-Senticons	Cant. SO Dictionaries	Coincid.	Coincid. sin concord.
Sustantivos	6.104	1.333	334	23
Adjetivos	3.757	2.052	824	117
Verbos	1.377	740	191	10
Adverbios	304	593	60	14

Tabla 5.2: Datos de los recursos utilizados para el etiquetado de polaridad.
Fuente: Elaboración Propia

La Tabla 5.2 reune toda la información asociada a estos recursos. Las dos últimas columnas corresponden a las palabras que se encuentran en ambos recursos y a las palabras que se encuentran en ambos recursos y que a la vez no concuerdan en la polaridad asignada.

Una vez que se realiza el etiquetado, se tendrá una cadena compuesta por etiquetas POS y NEG según corresponda. Luego, se utiliza la misma iteración utilizada en el atributo skipgrams para obtener los *polarity skipgrams* que componen a cada texto, como se ejemplifica en el Capítulo 4. Luego, las frecuencias de los bigramas posibles son reflejados en el vector de atributos. Al igual que en ocasiones anteriores, la función retorna un string con los valores del vector de atributos concatenados por comas.

5.3.6 Emotional Contexts

La última dimensión se compone de tres atributos: *Activation*, *Imagery* y *Pleasantness* pero su explicación se realiza en conjunto ya que para los tres atributos el procedimiento es exactamente igual. Incluso, es posible decir que el desarrollo de esta función es el más simple de todos las funciones desarrolladas.

1. **Activation, Imagery y Pleasantness**: Como se describe en el Capítulo 4, estos atributos son construidos en base a puntajes del 1 al 3. El primer paso es obtener una lista con todos los lemas de las palabras que componen el texto por lo que se realiza el pre-procesamiento básico descrito en oportunidades anteriores. Teniendo esta lista, el siguiente paso es obtener los puntajes asociados a cada lema y para esto se utiliza una adaptación del *Dictionary of Affect in Language* (DAL) al idioma Español, desarrollada por Gravano et al. (2013) en [67]. Esta tiene las mismas características del DAL original pero tiene una cantidad de 2.669 palabras, bastante menor a la cantidad del DAL original. El DAL en Español fue proporcionado en formato CSV, por lo que este archivo es leído por la función para asignar los puntajes a los lemas respectivos. Una vez asignados los puntajes, se obtiene el puntaje promedio para el texto completo.

El vector de atributos se define por un vector de tres valores: $[bajo, medio, alto]$. Cada valor es binario su valor por defecto es 0, el cual puede cambiar dependiendo de su puntaje promedio. El puntaje promedio es redondeado con la función *round*() de Python y se realiza una evaluación:

Si es igual a 1, $bajo = 1$

Si es igual a 2, $medio = 2$

Si es igual a 3, $alto = 3$

En el ejemplo dado en el Capítulo 4 donde los puntajes son 2.38, 2.37 y 2.23 para *Activation*, *Imagery* y *Pleasantness* respectivamente, los vectores de atributos resultantes serían:

Activation: $[0, 1, 0]$

Imagery: $[0, 1, 0]$

Pleasantness: $[0, 1, 0]$

Finalmente como en casos anteriores, los valores son concatenados utilizando comas y el string resultante es retornado por la función.

Esto concreta la descripción de todas las funciones que realizan la extracción de atributos. Para realizar el proceso completo, se utiliza un script en Python que utiliza todas las funciones implementadas y que luego inserta los valores obtenidos en el modelo de datos. La consulta SQL es idéntica para casi todos los casos (podría cambiar el nombre de un campo en el caso de las tablas *cgrams* y *skipgrams* debido a que tienen dos campos para sus vectores de atributos como se explicó anteriormente) por lo que su descripción fue dejada para el final. Ésta sería la siguiente:

```
INSERT INTO tabla (feature_vector, tweet_id) VALUES (var1, var2)
```

Donde $var1$ corresponde al string resultante de la concatenación de los valores del vector de atributos (excepto para los atributos *Counterfactual*, *Temporal Compression* y *Temporal Imbalance* donde son valores únicos) y $var2$ corresponde a la ID del tweet correspondiente en la tabla *tweet*. Para el caso del atributo *Contextual Imbalance* la consulta es distinta ya que el campo $feature\_vector$ se debe actualizar porque este campo tenía datos insertados con anterioridad producto de la primera parte de su desarrollo donde se obtienen las palabras que se utilizarán en el cálculo de la medida de similitud semántica. Entonces, para este caso la consulta es la siguiente:

```
UPDATE context_imbalance SET feature_vector = var1
WHERE tweet_id = var2
```

Donde *var*1 corresponde a su vector de atributos y *var*2 corresponde a la ID del tweet correspondiente en la tabla *tweet*.

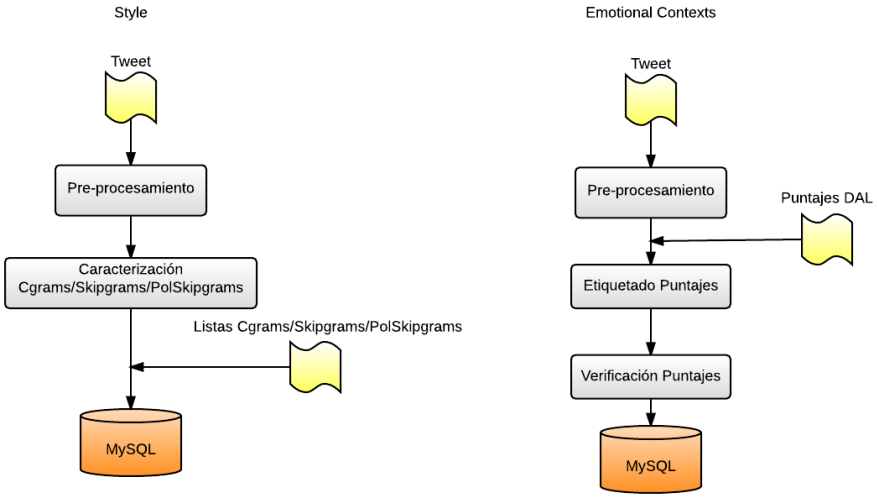

Figura 5.6: Esquema dimensiones Style y Emotional Contexts
Fuente: Elaboración Propia.

5.4 Módulo de Clasificación

La última etapa de esta implementación corresponde al módulo de clasificación, donde se realiza el entrenamiento y las pruebas del algoritmo de clasificación. Para el desarrollo de este módulo se utilizan dos herramientas principales. La primera corresponde al paquete *scikit-learn* para Python, el cual contiene una gran variedad de herramientas asociadas a Machine Learning. La otra herramienta corresponde a *NumPy*, la cual es requerida por scikit-learn para su uso.

Para dar introducción a estos procedimientos, a continuación se describen los sets de entrenamiento y pruebas utilizados. Este estudio se compondrá de dos pruebas distintas, por lo que se construirán dos sets de entrenamiento y dos sets de pruebas. En la primera prueba se realizará una clasificación de textos irónicos respecto a un set compuesto por textos etiquetados como irónicos durante la evaluación descrita en el Capítulo 3 (set positivo) y además por textos extraídos desde medios de prensa (set negativo). En cambio, en la segunda prueba se realizará una clasificación donde el set negativo estará compuesto por aquellos textos etiquetados como no irónicos durante las evaluaciones.

	Prueba 1	Prueba 2
Set positivo	9.000	7.000
Set negativo	9.000	7.000

Tabla 5.3: Cantidad de datos positivos y negativos distribuidos para las pruebas.
Fuente: Elaboración Propia

La Tabla 5.3 contiene las cantidades de los tipos de datos que son usados para cada prueba. Se optó por utilizar sets equilibrados respecto a la cantidad de datos positivos y negativos.

El set correspondiente a la prueba 2 tiene menos datos que el set para la prueba 1 ya que se decidió que el límite fuera determinado por la cantidad de datos negativos existentes, los cuales son cercanos a los 7.000 como se detalla en el Capítulo 3.

5.4.1 Entrenamiento, Pruebas y Evaluación

Para comenzar con el entrenamiento, el primer paso es recolectar los vectores de atributos. Para generar los sets de datos, se obtienen al azar las ID de los tweets que cumplan con las características necesarias. Para la prueba 1 se exige que los tweets del set positivo sea cualquier tweet que tenga un valor $label = 1$ y para el set negativo se exige que sea cualquier tweet con un valor $label = 0$ pero además que id>17277 para seleccionar solo aquellos datos negativos extraídos desde medios de prensa. En cambio, para la prueba 2 el set positivo cumple con las mismas características pero el set negativo exige un valor $label = 0$ y un valor id<17278 para seleccionar solo aquellos datos negativos generados a través de la evaluación con personas.

Una vez que se tienen estas listas, el siguiente paso es iterar sobre ellas para obtener los vectores de atributos y esto se consigue a través de la siguiente consulta SQL:

```
SELECT T.label, PT.feature_vector, CF.feature_vector, TC.feature_vector,
CI.feature_vector, TI.feature_vector, A.feature_vector, I.feature_vector,
P.feature_vector, PS.feature_vector, SG.feature_vector, SG.feature_vector
FROM tweet T, pointedness PT, counterfact CF, temp_comp TC, context_imbalance CI,
temp_imbalance TI, activation A, imagery I, pleasantness P, polarity_skipgrams PS,
skipgrams SG, cgrams CG
WHERE T.id = var1
AND T.id = PT.id
AND PT.id = CF.id
AND CF.id = TC.id
AND TC.id = CI.id
AND CI.id = TI.id
AND TI.id = A.id
AND A.id = I.id
AND I.id = P.id
AND P.id = PS.id
AND PS.id = SG.id
AND SG.id = CG.id
```

Donde $var1$ corresponde a la ID de un tweet, el cual va cambiando en cada iteración. Luego, cada vector de atributos obtenido (los cuales corresponden a strings) son concatenados por comas como se ha realizado en oportunidades anteriores. Finalmente, este string de números enteros concatenados por comas es transformado en una lista (arreglos en Python) utilizando la función $split(',')$. Con este tratamiento, para cada tweet se tendrá un vector de atributos final el cual representa todos los atributos para este tweet.

El siguiente paso corresponde a tratar los datos y prepararlos para el proceso de entrenamiento, lo que equivale a transformar estos datos en matrices de *NumPy*. El algoritmo recibe dos tipos de datos para su entrenamiento. El primero corresponde a un vector de dimensiones $n \times 1$ donde n corresponde a la cantidad de datos en el set de entrenamiento. Este vector corresponde a las etiquetas de las distintas clases que se pretende clasificar, para cada uno de los sujetos evaluados y generalmente se le llama Y. El segundo corresponde a una matriz de dimensiones $n \times m$ donde n corresponde a la cantidad de sujetos evaluados y m corresponde a la cantidad de atributos que caracterizan a estos sujetos. Esta matriz es llamada X.

Para su construcción, primero se crean vectores y matrices vacíos utilizando la función de NumPy *zeros*() la cual crea una matriz de ceros según las dimensiones dadas. Finalmente, se itera sobre las listas creadas anteriormente, se separa el primer valor de los demás ya que corresponde a la etiqueta que determina la clase del tweet (debido a cómo fue seleccionada por la consulta SQL anterior) y finalmente se van agregando los datos a sus matrices correspondientes utilizando la función de NumPy *array*(). Luego, una vez que se tienen X e Y preparados, los datos son divididos en proporción $70 - 30$ utilizando un 70% de los datos para el proceso de entrenamiento y un 30% para el proceso de pruebas. La división se realiza de tal forma, que tanto la porción de entrenamiento como la porción de pruebas, tenga la misma cantidad de datos positivos y negativos.

Los algoritmos de clasificación utilizados son incluidos en el paquete scikit-learn por lo que no se requiere una mayor implementación en este paso. Para utilizar el método de clasificación *Naive Bayes Multinomial* se utiliza la función $MultinomialNB()$ que se encuentra dentro del módulo *naive_bayes* de scikit-learn. Esto define un objeto que caracteriza al algoritmo clasificador, el cual es entrenado por medio de la función $fit()$ cuyos parámetros corresponden a las porciones de entrenamiento de X e Y respectivamente.

Para realizar el procedimiento de prueba, se utiliza la función *predict*() la cual utiliza como parámetro la porción de pruebas de X. A cada texto representado en esta matriz, le asignará una etiqueta de predicción las cuales serán almacenadas en una variable $Y\_predict$, el cual corresponde a un vector con las mismas dimensiones que la porción de pruebas de Y. Finalmente, para obtener las medidas de evaluación de resultados, se utiliza el módulo *metrics* el cual incluye las funciones *accuracy_score*(), *precision_score*(), *recall_score*() y $f1\_score()$. Los parámetros que reciben estas funciones corresponden a la porción de pruebas de Y y a $Y\_predict$.

Por último, como última evaluación se realiza el procedimiento *10-fold Cross Validation* el cual también se encuentra implementado en el paquete scikit-learn. Para su uso, se debe disponer del módulo *cross-validation* y utilizar la función $KFold$ cuyos parámetros corresponden al largo del vector Y y a k que en este caso es igual a 10. Esta función da como resultado un objeto sobre el cual se puede iterar y en cada iteración tiene dos valores, correspondientes a los índices asociados a X e Y en cada iteración del proceso. Cabe mencionar que antes de realizar este procedimiento, X e Y son distribuídos al azar (respetando la correspondencia entre ambos) utilizando la función *shuffle* de NumPy.

Capítulo 6

Resultados

El siguiente capítulo muestra los resultados obtenidos a través del proceso de extracción de atributos y la clasificación de tweets irónicos respecto a un set de datos negativo. Para cada atributo extraído, se presentan las cantidad de patrones identificados por set de datos y algunos de los patrones con un nivel de frecuencia significativa en cada set. Respecto al proceso de clasificación, se muestra el nivel de *accuracy* que se obtiene al agregar los atributos del modelo en forma progresiva para caracterizar a los textos y luego se muestran los resultados generales para ambas pruebas junto con los resultados del comportamiento de la clasificación a través de un proceso 10-Fold Cross Validation.

6.1 Extracción de Atributos

La extracción de atributos fue realizado con éxito a través de las dimensiones del modelo. El procedimiento de extracción para los datos que componen el corpus tardó aproximadamente 72 horas en completarse, siendo realizado de forma automática y continua en un computador de gama alta utilizando las herramientas descritas en el Capítulo 5 bajo el sistema operativo *Windows 8*. A continuación, se darán a conocer algunos datos y observaciones respecto a los resultados del proceso.

1. **Signatures**: En esta dimensión es posible resumir los siguientes datos. El atributo *Pointedness* se compone de tres valores asociados a *emoticonos*, *signos de puntuación* y *palabras en mayúsculas*. La Tabla 6.1 resume la detección del atributo *Pointedness*. Como se aprecia en la tabla, *set evaluado* corresponde al set de datos recolectados en primera instancia y evaluados con personas (el cual contiene casos positivos y negativos), mientras que *set prensa* corresponde al set negativo construido en base a cuentas de Twitter de medios de prensa (el cual solo contiene casos negativos). El set evaluado positivo contiene 9.812 tweets, el set evaluado negativo contiene 7.465 tweets y finalmente el set prensa contiene 17.855 tweets. Estos nombres se mantienen para el resto de las tablas y figuras. La Tabla 6.2 resume la detección de los atributos *Counterfactual* y *Temporal Compression*.

	Emoticonos	Puntuación	Mayúsculas
Set evaluado (pos)	1.033 (10,53%)	7.807 (79,57%)	1.611 (16,42%)
Set evaluado (neg)	661 (8,85%)	5.727 (76,72%)	1.310 (17,55%)
Set prensa	43 (0,24%)	8.519 (47,71%)	3.324 (18,62%)

Tabla 6.1: Detección de patrones del atributo *Pointedness*.
Fuente: Elaboración Propia

	Counterfactual	Temporal Compression
Set evaluado (pos)	1.892 (19,28%)	898 (9,15%)
Set evaluado (neg)	1.270 (17,01%)	531 (7,11%)
Set prensa	1.194 (6,69%)	389 (2,17%)

Tabla 6.2: Detección de patrones de los atributos *Counterfactual* y *Temporal Compression*.
Fuente: Elaboración Propia

Se aprecia en la Tabla 6.1 que el atributo más presente en los tres sets corresponde a los signos de puntuación, algo esperable dados los patrones utilizados para esta detección. Además, se observa que los emoticonos representan una cantidad mayor del set evaluado positivo respecto a lo que representa para los otros sets, sobretodo en comparación al set prensa. En la Tabla 6.2 se aprecia que hay una mayor frecuencia de los adverbios seleccionados en el set evaluado positivo respecto a los demás, pero esta diferencia no es tan grande comparándolo con el set evaluado negativo. Respecto a estas identificaciones, la Tabla 6.3 resume los 10 emoticonos más frecuentes en los distintos sets, junto a su frecuencia y de forma análoga, la tabla 6.4 resume los 10 adverbios más utilizados en cada set y para cada uno de los atributos *Counterfactual* y *Temporal Compression*.

Set evaluado (pos)	Set evaluado (neg)	Set prensa
:) (165)	xD (92)	:/ (13)
¬¬ (152)	:) (92)	S: (12)
:D (132)	:D (66)	:) (5)
-.- (104)	-.- (46)	:D (4)
xD (97)	¬¬ (44)	:P (3)
:((53)	:((41)	:L (3)
:/ (42)	XD (32)	(8 (2)
;) (37)	:S (32)	;) (1)
:S (34)	:/ (30)	-
XD (24)	;) (22)	-

Tabla 6.3: Emoticonos más frecuentes detectados en el atributo *Pointedness*.
Fuente: Elaboración Propia

Counterfactual			Temp. Compression		
Set ev.(pos)	Set ev.(neg)	Set prensa	Set ev.(pos)	Set ev.(neg)	Set prensa
mas (690)	ahora (408)	más (734)	ahora (311)	ahora (204)	desde (178)
ahora (622)	mas (373)	menos (135)	mañana (295)	mañana (107)	ahora (46)
más (387)	pero (298)	pero (96)	luego (78)	pronto (62)	después (42)
pero (383)	más (241)	ahora (92)	pronto (59)	desde (36)	mañana (42)
menos (160)	menos (94)	sólo (69)	después (58)	ayer (36)	luego (40)
casi (55)	casi (46)	cerca (66)	desde (58)	después (35)	recién (11)
aun (54)	solo (29)	casi (41)	ayer (39)	luego (34)	ayer (11)
sólo (45)	sólo (27)	aún (30)	rápido (23)	rápido (16)	pronto (9)
solo (43)	cerca (19)	a pesar de (7)	mas tarde (6)	de repente (7)	en breve (5)
aún (23)	aun (18)	por lo tanto (5)	de repente (6)	recién (5)	rápido (5)

Tabla 6.4: Palabras más frecuentes detectadas en los atributos *Counterfactual* y *Temporal Compression*.
Fuente: Elaboración Propia

2. **Unexpectedness**: Para esta dimensión es posible realizar algunas observaciones. La Tabla 6.5 resume las frecuencias de la extracción del atributo *Contextual Imbalance* y la Tabla 6.6 resume el atributo *Temporal Imbalance*.

	Set ev. (pos)	Set ev. (neg)	Set Prensa
Min	80 (0,82%)	103 (1,38%)	138 (0,77%)
Med	2.324 (23,69%)	1.873 (25,09%)	6.127 (34,32%)
Max	7.408 (75,5%)	5.489 (73,53%)	11.590 (64,91%)

Tabla 6.5: Detección de patrones del atributo *Contextual Imbalance*.
Fuente: Elaboración Propia

A diferencia de lo que podría indicar la teoría detrás del atributo *Contextual Imbalance*, no se identifican diferencias significativas entre los distintos sets. A pesar de que en el set evaluado positivo, existe una mayor presencia de medidas máximas (es decir, con un desequilibrio contextual muy alto) respecto a lo apreciado en el set prensa, comparado al set evaluado positivo la diferencia es ínfima. Esto puede deberse a las limitaciones asociadas a las herramientas utilizadas, que no son tan completas como las existentes en Inglés. El cálculo de una medida de similitud semántica en Español es un recurso casi inexistente dentro de las herramientas de Análisis de Sentimientos actuales, el cual solo ha sido desarrollado a nivel experimental.

	Set ev. (pos)	Set ev. (neg)	Set Prensa
Temporal Imbalance	26 (0,26%)	22 (0,29%)	49 (0,27%)

Tabla 6.6: Detección de patrones del atributo *Temporal Imbalance*.
Fuente: Elaboración Propia

En el caso del atributo *Temporal Imbalance* el desempeño es insuficiente. El problema es similar al descrito anteriormente. Los recursos lingüísticos en Español para evaluar automáticamente la polaridad de una palabra es un recursos limitado, escaso y no tan completo como los existentes en otros idiomas. Por este motivo, existe una gran

cantidad que verbos que no son encontrados en las listas utilizadas para el etiquetado y por ende, no contienen su etiqueta y no son consideradas por el proceso de extracción.

3. **Style**: En esta dimensión, los vectores son representaciones del tipo *Bag-of-Words* de cada texto, como se discute en el Capítulo 5. Los gráficos en las Figuras 6.1, 6.2, 6.3, 6.4 y 6.5 muestran los patrones más frecuentes de todos los encontrados. Se utilizan gráficos distintos para las dos pruebas realizadas ya que el vector cambia dependiendo del set de entrenamiento utilizado.

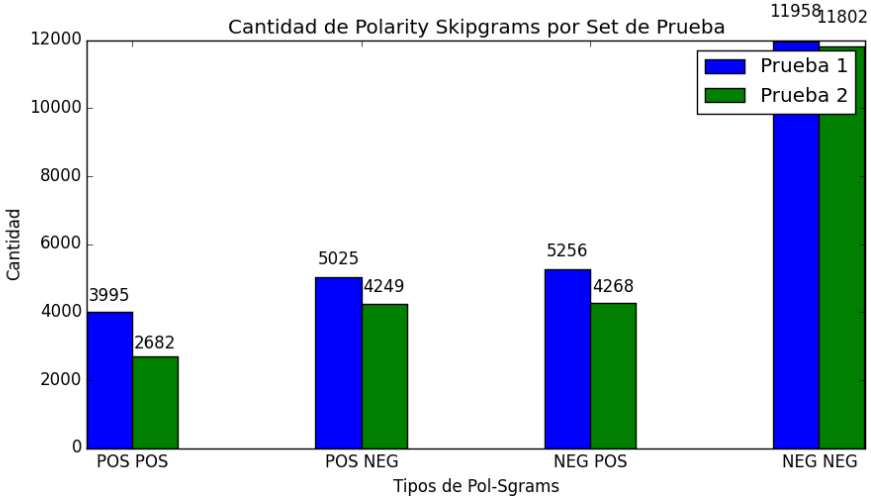

Figura 6.1: Frecuencia de Polarity Skipgrams encontrados en Pruebas 1 y 2
Fuente: Elaboración propia.

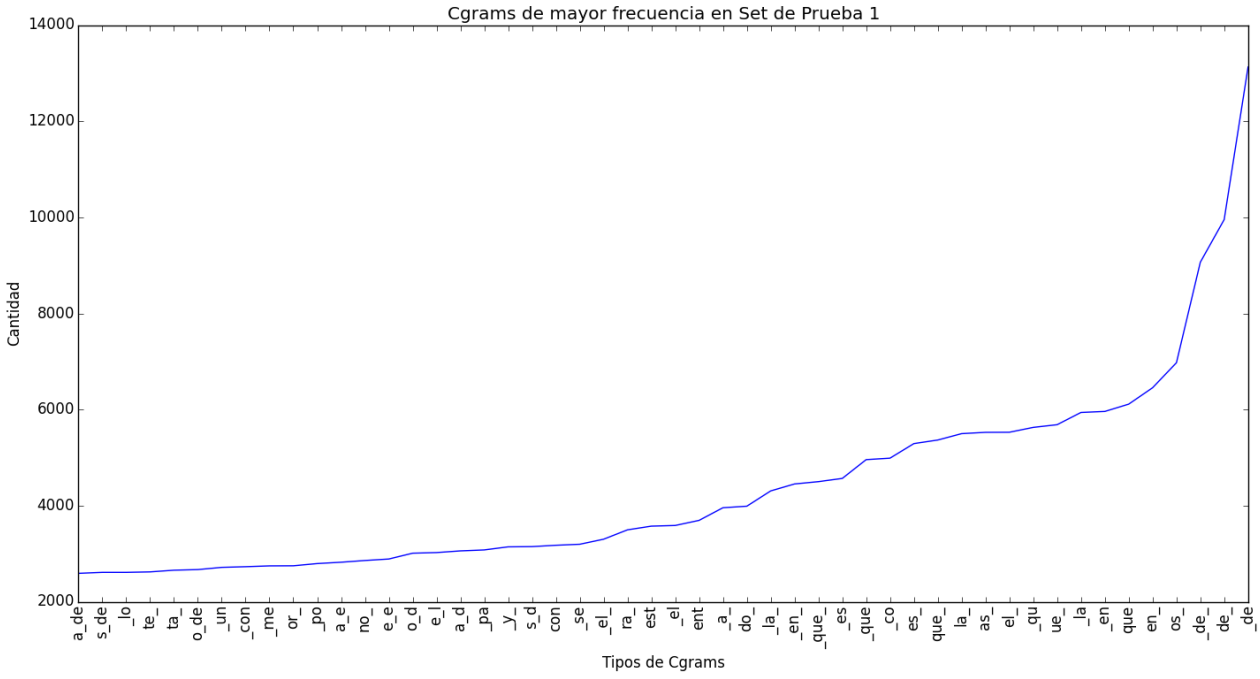

Figura 6.2: 50 cgrams más frecuentes encontrados en Prueba 1
Fuente: Elaboración propia.

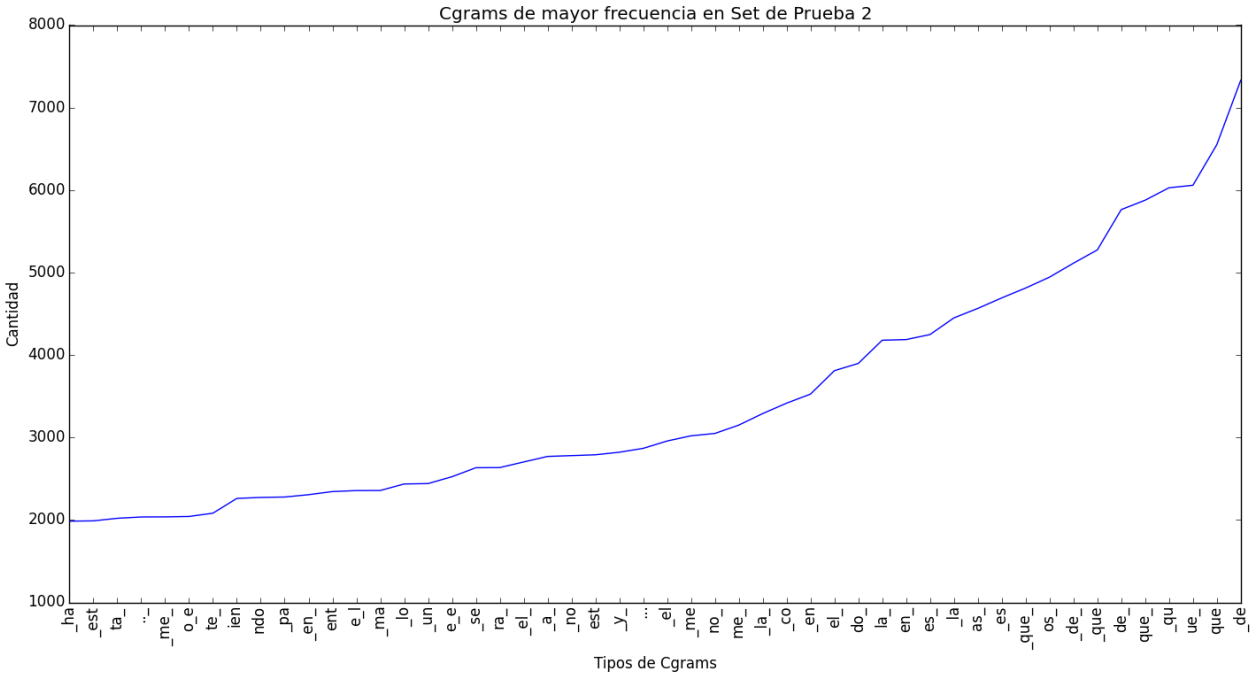

Figura 6.3: 50 cgrams más frecuentes encontrados en Prueba 2
Fuente: Elaboración propia.

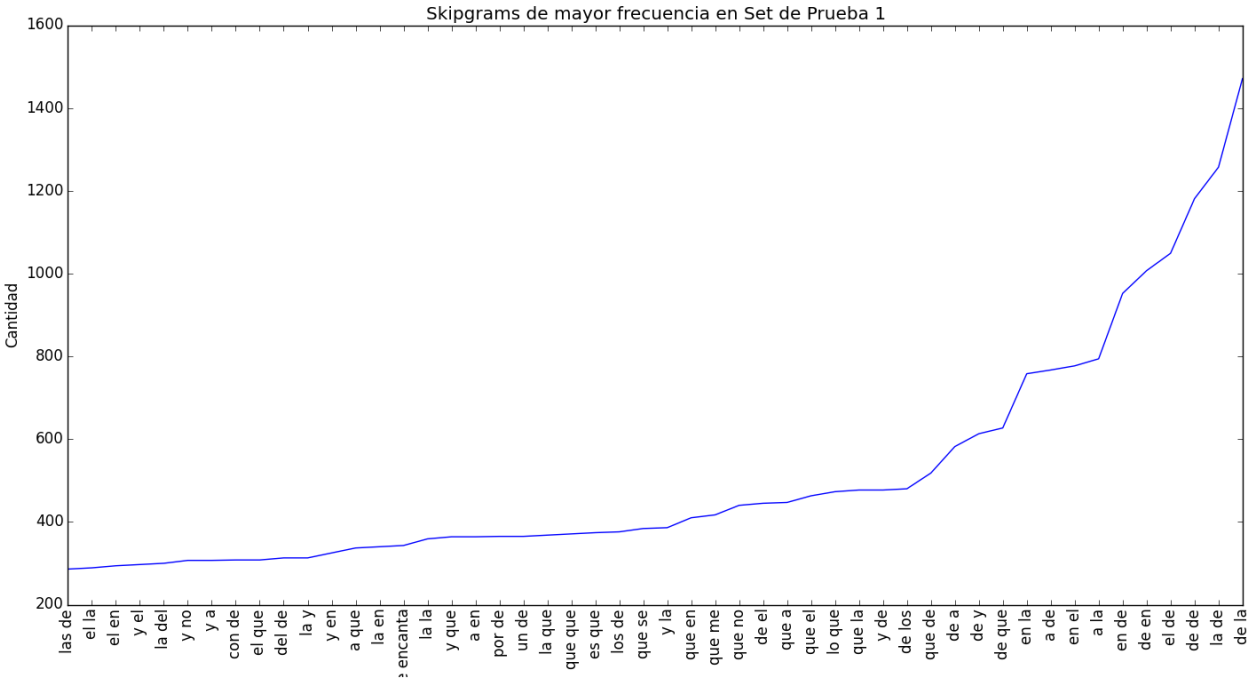

Figura 6.4: 50 Skipgrams más frecuentes encontrados en Prueba 1
Fuente: Elaboración propia.

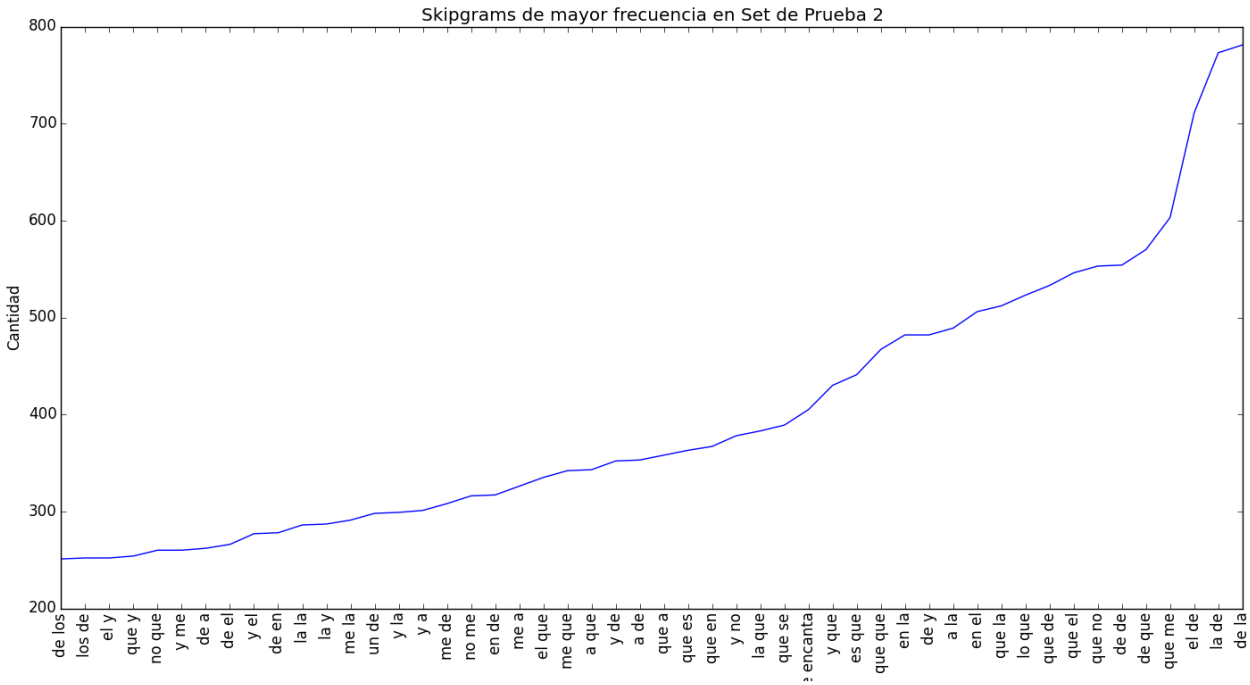

Figura 6.5: 50 Skipgrams más frecuentes encontrados en Prueba 2
Fuente: Elaboración propia.

Respecto a los atributos *Character N-grams* y *Skipgrams* se tienen dos gráficos, uno respecto al set de entrenamiento de cada prueba. Los vectores para estos atributos se construyen en base a cada set de entrenamiento, como se explica en el Capítulo 5. El atributo *Polarity Skipgrams* también es afectado por la naturaleza de los recursos lingüísticos utilizados, pero el impacto no es tan drástico como en el atributo *Temporal Imbalance* ya que este atributo también busca otros tipos de palabras más allá de verbos por lo que para este atributo, los recursos son más funcionales.

4. **Emotional Context**: Por último se tienen los valores caracterizados por los atributos *Activation*, *Imagery* y *Pleasantness*, reflejando un resumen en las Tablas 6.7, 6.8 y 6.9 respectivamente. A lo largo de estos tres atributos no se observan diferencias significativas entre los distintos sets, lo que puede deberse a las características del DAL en Español. Este recurso también es limitado respecto a su contraparte en Inglés al contener una cantidad mucho menor de palabras evaluadas, lo que podría sesgar el nivel de información que cada promedio transmite.

	Set ev. (pos)	Set ev. (neg)	Set Prensa
Bajo	19 (0,19%)	28 (0,38%)	41 (0,23%)
Medio	5.963 (60,77%)	4.147 (55,55%)	10.533 (58,99%)
Alto	3.612 (36,81%)	3.038 (40,69%)	6.832 (38,26%)

Tabla 6.7: Detección de patrones del atributo *Activation*.
Fuente: Elaboración Propia

	Set ev. (pos)	Set ev. (neg)	Set Prensa
Bajo	9 (0,092%)	6 (0,08%)	45 (0,25%)
Medio	4.696 (47,86%)	3.648 (48,87%)	7.437 (41,65%)
Alto	4.889 (49,83%)	3.559 (47,68%)	9.924 (55,58%)

Tabla 6.8: Detección de patrones del atributo *Imagery*.
Fuente: Elaboración Propia

	Set ev. (pos)	Set ev. (neg)	Set Prensa
Bajo	55 (0,56%)	55 (0,74%)	345 (1,93%)
Medio	6.334 (64,55%)	4.629 (62,01%)	13.699 (76,72%)
Alto	3.205 (32,66%)	2.529 (33,88%)	3.362 (18,82%)

Tabla 6.9: Detección de patrones del atributo *Pleasantness*.
Fuente: Elaboración Propia

6.2 Clasificación

A continuación, se muestra el detalle de los resultados del proceso de pruebas de clasificación utilizando el método Naive Bayes Multinomial, luego de realizar los entrenamientos correspondientes como se describe en el Capítulo 5. Como se ha mencionado en oportunidades anteriores, se realizan dos pruebas; la primera utiliza como set negativo aquellos textos extraídos desde cuentas de Twitter de prensa, y la segunda utiliza como set negativo los textos que fueron etiquetados utilizando evaluadores humanos. Los sets para ambas pruebas son equilibrados, contiendo la misma cantidad de casos positivos y negativos. Por este motivo, la *línea de base* para las medidas de *accuracy* corresponde al 50%. El proceso de pruebas se realizó agregando los atributos de forma progresiva, partiendo por el atributo *Pointedness* y terminando con el atributo *Character N-grams*. La Tabla 6.10 muestra los niveles de *accuracy* obtenidos en ambas pruebas para cada atributo, a medida que éstos se van agregando de forma progresiva partiendo desde *Pointedness* hasta el atributo *Character N-grams*.

	Prueba 1	Prueba 2
Pointedness	67,22%	50,66%
Counterfactual	60,04%	50,55%
Temporal Compression	60,94%	50,86%
Contextual Imbalance	66,07%	53,02%
Temporal Imbalance	66,05%	53,05%
Activation	66,35%	53,95%
Imagery	66,22%	54,38%
Pleasantness	68,44%	53,76%
Polarity Skipgrams	66,46%	54,07%
Skipgrams	91,33%	56,33%
Cgrams	95,25%	58,26%

Tabla 6.10: Niveles de Accuracy obtenidos al agregar atributos en forma progresiva para ambas pruebas
Fuente: Elaboración Propia

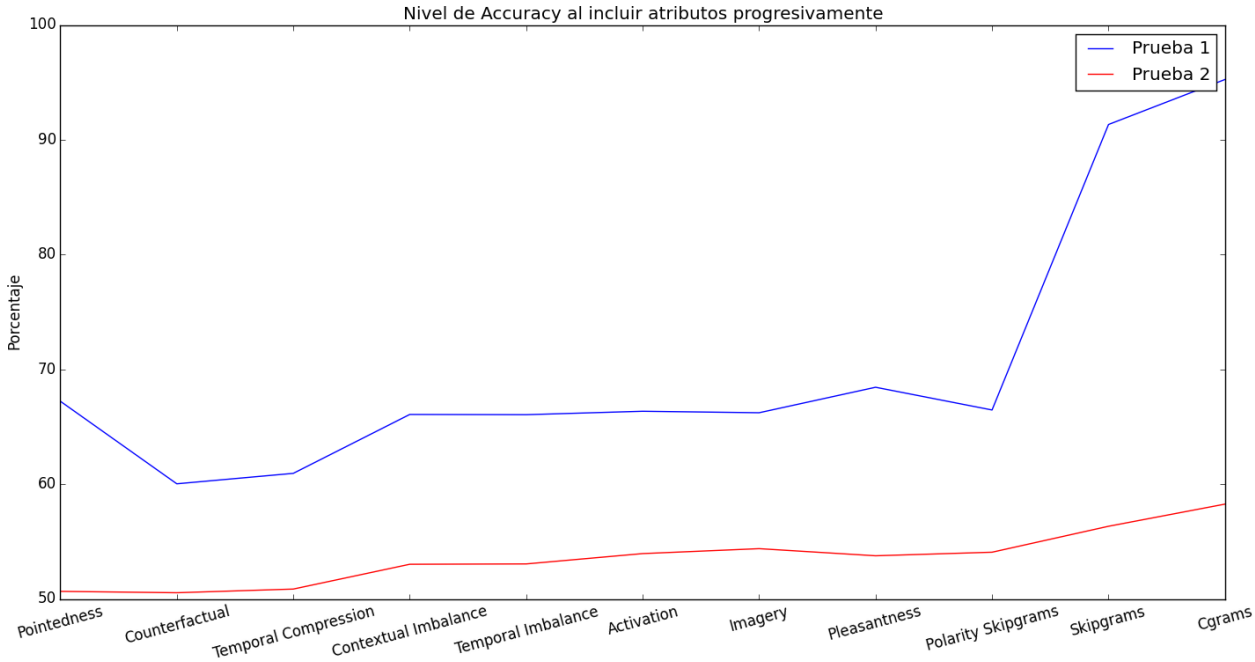

Figura 6.6: Gráfico de niveles de Accuracy obtenidos al agregar atributos en forma progresiva para ambas pruebas
Fuente: Elaboración propia.

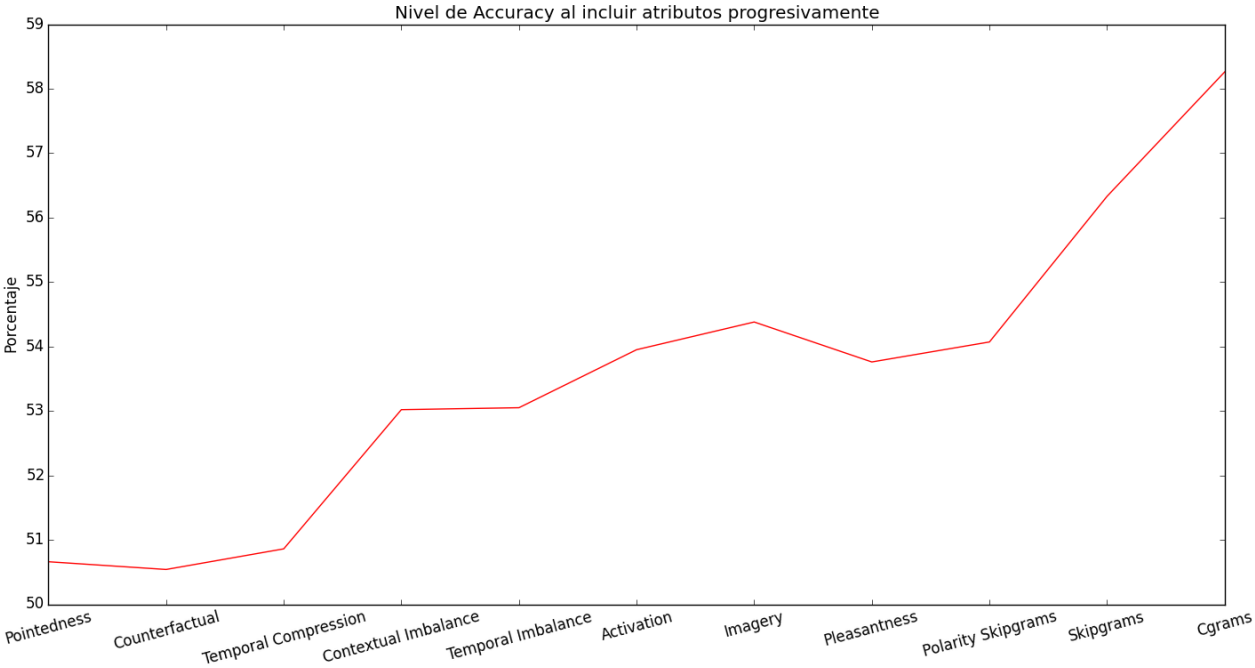

Figura 6.7: Gráfico de niveles de Accuracy obtenidos al agregar atributos en forma progresiva para Prueba 2
Fuente: Elaboración propia.

Los datos muestran que de todos los atributos, los correspondientes a la dimensión *Style* son los que aportan el mayor nivel de *accuracy*. Existen atributos que aportan de forma negativa a los resultados, pero este comportamiento puede variar dependiendo del set de entrenamiento. El único que reduce el nivel de *accuracy* en ambas pruebas es el atributo *Counterfactual*.

1. **Prueba 1**: A continuación, se muestran los resultados finales de la Prueba 1 resumidos en la Tabla 6.11. Los niveles de todas las medidas de evaluación superan el 90%, lo que representan buenos resultados. Esto indica que el modelo es capaz de diferenciar textos irónicos de texto más ligado a contenido objetivo que subjetivo. Sin embargo, cabe recalcar que esto se debe en gran parte a los atributos de análisis sintáctico, más que por los atributos que analizan el texto a nivel semántico. Esto se debe en parte a las debilidades asociadas a las herramientas para el idioma Español, pero hay otras consideraciones que serán discutidas posteriormente.

   El proceso de 10-Fold Cross Validation muestra consistencia con la prueba realizada en el set original, teniendo valores de *accuracy*, *precision*, *recall* y *F-measure* muy similares en cada iteración.

	Resultados Prueba 1
Accuracy	95,26%
Precision	95,97%
Recall	94,48%
F-measure	95,22%

Tabla 6.11: Resultados finales de Prueba 1
Fuente: Elaboración Propia

	10-Fold Cross Validation Prueba 1			
	Accuracy	Precision	Recall	F-measure
Fold 1	95,11%	95,29%	94,86%	95,07%
Fold 2	95,06%	95,88%	94,04%	94,95%
Fold 3	95,17%	96,06%	93,99%	95,01%
Fold 4	94,5%	95,02%	93,85%	94,44%
Fold 5	95,17%	95,99%	94,16%	95,07%
Fold 6	95,66%	96,69%	94,53%	95,59%
Fold 7	95,72%	95,92%	95,38%	95,65%
Fold 8	95,44%	95.94%	94,87%	95,39%
Fold 9	95,17%	96,11%	94,58%	95,34%
Fold 10	95,06%	95,84%	94,49%	95,17%
Promedio	95,21%	95,87%	94,48%	95,17%

Tabla 6.12: Resultados de 10-Fold Cross Validation de Prueba 1
Fuente: Elaboración Propia

2. **Prueba 2**: A diferencia de los resultados obtenidos en la Prueba 1, donde los niveles de *accuracy* superan en un 40% a la línea de base, los resultados de la Prueba 2 apenas superan la línea de base en un 8% aproximadamente. Esta comparación deja en evidencia que el comportamiento del modelo depende de la naturaleza del set negativo utilizado, algo similar a lo que muestra el modelo original en [5]. El nivel de *recall* es el más alto de todas las medidas de evaluación, indicando que a pesar del bajo desempeño, el modelo es capaz de clasificar correctamente un 60% de los casos positivos en el set de prueba. Es decir, el modelo identifica algunos textos irónicos de forma correcta a pesar de que el desempeño total no sea satisfactorio en este caso.

	Resultados Prueba 2
Accuracy	58,26%
Precision	57,98%
Recall	60%
F-measure	58,98%

Tabla 6.13: Resultados finales de Prueba 2
Fuente: Elaboración Propia

	10-Fold Cross Validation Prueba 2			
	Accuracy	Precision	Recall	F-measure
Fold 1	60,86%	59,77%	61,17%	60,46%
Fold 2	58,43%	55,54%	60,57%	57,95%
Fold 3	59,07%	60,17%	58,74%	59,45%
Fold 4	60,5%	60,14%	60,58%	60,36%
Fold 5	58,14%	57,57%	60,55%	59,02%
Fold 6	56,36%	55,92%	58,88%	57,36%
Fold 7	57,5%	57,36%	58,92%	58,13%
Fold 8	60,43%	61,22%	59,66%	60,43%
Fold 9	58%	60,11%	61,08%	60,59%
Fold 10	58,64%	57,96%	62,52%	60,15%
Promedio	58,79%	58,58%	60,27%	59,39%

Tabla 6.14: Resultados de 10-Fold Cross Validation de Prueba 2
Fuente: Elaboración Propia

A partir de las pruebas señaladas, es posible observar que el desempeño del modelo es dependiente del set negativo utilizado, algo similar a lo que ocurre en la implementación original [5], sin embargo, el desempeño mostrado acá en una de las pruebas es peor que los resultados más bajos obtenidos en la implementación original utilizando un set de entrenamiento y pruebas equilibrado. En el éxito asociado a la Prueba 1 y las falencias de esta prueba, influyen una serie de detalles desde la naturaleza de los datos hasta las limitaciones de las herramientas, los cuales son analizados en la discusión posterior y en el Capítulo 7.

6.3 Discusión

En función de los resultados obtenidos, es posible realizar un análisis crítico respecto a la metodología seguida y a las herramientas que fueron utilizadas para la implementación de este trabajo. Para complementar este análisis, a continuación se muestran los resultados finales (aproximados) de las pruebas 1 y 2 a través de matrices de confusión, calculadas a partir de los valores de *accuracy*, *precision* y *recall* obtenidos. Se recuerda que, según lo descrito en el Capítulo 5, para ambas pruebas se utilizaron sets equilibrados (es decir, con la misma cantidad de casos positivos y negativos) utilizando 18.000 tweets para la Prueba 1 y 14.000 tweets para la Prueba 2.

		Clases Predecidas	
		Irónico	No Irónico
Clases Reales	Irónico	8.503 (47,24%)	497 (2,76%)
	No Irónico	356 (1,98%)	8.644 (48,02%)

Tabla 6.15: Resultados de Prueba 1 desglosados en una Matriz de Confusión.
Fuente: Elaboración Propia

		Clases Predecidas	
		Irónico	No Irónico
Clases Reales	Irónico	4.200 (30%)	2.800 (20%)
	No Irónico	3.044 (21,74%)	3.956 (28,26%)

Tabla 6.16: Resultados de Prueba 2 desglosados en una Matriz de Confusión.
Fuente: Elaboración Propia

6.3.1 Construcción del Corpus

Uno de los elementos fundamentales dentro del desarrollo del clasificador corresponde al corpus de textos irónicos y no irónicos. Como se indica en el Capítulo 3, se utilizó inicialmente un método semiautomático para su construcción (en base al hashtag *#Ironía* y sus símiles) para luego corroborar el etiquetado de cada texto, utilizando evaluadores humanos. La calidad del corpus impacta directamente en los resultados de un clasificador de texto como el utilizado en este trabajo, ya que a partir de éste, se le *enseña* a la máquina a diferenciar los tipos de textos que se busca caracterizar.

Si bien el método de utilizar evaluadores humanos complementa de forma positiva el método semiautomático seguido en [5], su efecto puede ser además contraproducente. La ironía es un fenómeno complejo de identificar e interpretar para una persona, inclusive, existen estudios que muestran que la comprensión de una expresión irónica requiere una capacidad cognitiva avanzada, propia de un adulto [68] o también, que su comprensión requiere de funciones cognitivas tales, que para un humano con cierto tipo de daño neurológico, sería una tarea mucho más complicada de realizar [69]. Considerando esto, no se puede asegurar que una persona común y corriente sea capaz de detectar ironía con absoluta certeza mediante la lectura de un texto. Entonces, al tener solo 3 evaluadores como máximo para cada texto, es de esperar que la evaluación no sea del todo aceptable (esto queda parcialmente en evidencia

ante los resultados del estadístico *kappa de Cohen* mostrados en el Capítulo 3). En síntesis, la evaluación realizada con humanos en este trabajo, no es suficiente para generar un corpus *universal* de detección de ironía (es decir, un corpus que permita detectar ironía en cualquier escenario de comparación).

Sin embargo, dadas las características del método semiautomático, se considera que el impacto negativo es parcial. Como se menciona en [32], detrás de la etiqueta aplicada por el autor de un texto, existe una intención de comunicar algo. En este caso, ese algo es ironía, por lo que existe una alta probabilidad de que los textos extraídos desde Twitter con ese hashtag, hayan tenido la intención de transmitir un mensaje irónico. Debido a esto, es más probable que el set de textos evaluados como irónicos (set positivo), sea de mejor calidad que el set de textos evaluados como no irónicos (set negativo). En otras palabras, debido al origen de los datos, es más probable que el etiquetado de textos irónicos sea más certero que el etiquetado de textos no irónicos. Esto se ve reflejado en los resultados de las pruebas, ya que el clasificador es más preciso al diferenciar el set positivo versus textos con un origen y naturaliza diferentes, que al diferenciar el set positivo versus el set negativo, los cuales tienen un origen y naturaleza muy similares.

La diferencia que existe entre las pruebas 1 y 2 muestran que el método de construcción utilizado para generar el corpus no es suficiente. La cantidad de evaluadores humanos utilizados no permite obtener una evaluación representativa suficiente para filtrar de forma óptima los resultados del método semiautomático. Es necesario realizar una evaluación con una cantidad masiva de personas para caracterizar distintas formas de ironía en Español, presentes en Twitter. Esto puede ser posible a través de herramientas disponibles a través de la Web como *Mechanical Turk* de Amazon[1], donde una persona puede poner a disposición una tarea simple que requiere ser desarrollada por un humano, para que luego ésta sea realizada por personas de todo el mundo, a cambio de una remuneración.

Además, respecto a la construcción de un set negativo, cabe mencionar que el universo de textos que pueden representar el hecho de ser *no irónico* es mucho más extenso que aquellos que están contenidos dentro de los textos obtenidos a través del hashtag *#Ironía* y sus derivados. Es decir, el supuesto detrás del método semiautomático tal vez sirva para la construcción de un set positivo de datos irónicos, pero no basta para construir un set negativo de datos que permita detectar ironía en un texto inmerso de cualquier contexto.

El fenómeno de la ironía es amplio a su vez. Puede variar en la forma en que se expresa, depender del contexto, depender de situaciones específicas (por ejemplo, cuando se hacen referencias irónicas a medidas políticas en un país y gobierno determinados) e incluso pueden depender del país y el idioma en que se expresa. Por esto, tal vez buscar una solución universal para clasificar este fenómeno de forma automática, sea un camino equivocado. Es necesario investigar respecto a el desempeño que podría tener un clasificador de ironía en texto, en el cual se considere la incidencia de las características de su comunicación, presentes en un ambiente local. Por ejemplo, ¿Existen frases, signos de puntuación u otros símbolos utilizados con frecuencia para transmitir ironía en un país en específico? ¿Existen palabras, abreviaciones que la gente utilice más que otras cuando es irónica en este país?. Este trabajo

[1]https://www.mturk.com/mturk/welcome

consideró solo textos que estuviesen en Español para la construcción del corpus más otras características identificables independiente del idioma, pero tal vez si se centraran los esfuerzos en un solo contexto sociocultural, para caracterizar la ironía respecto a la idiosincrasia de un país o zona del mundo en específico, se podrían obtener resultados más acertados. En otras palabras, se sugiere considerar *Dividir para reinar* y pensar en obtener soluciones locales, pero más efectivas.

6.3.2 Modelo y Herramientas utilizadas

Respecto a las características del modelo y el comportamiento de sus atributos, se puede decir que según lo que muestran los resultados en la Tabla 6.10, no todos los atributos aportan cantidades significativas de información a la clasificación. El modelo consta de atributos que caracterizan elementos sintácticos (Pointedness, Counterfactual, Temporal Compression, Skipgrams y Cgrams), elementos semánticos (Contextual Imbalance y Temporal Imbalance), combinaciones de elementos sintácticos y semánticos (Polarity Skipgrams) y elementos asociados a la percepción y respuesta psicológica de las personas en función de las palabras que leen (Activation, Imagery y Pleasantness) pero de todos estos, se aprecia que dentro de aquellos cercanos a los elementos sintácticos, están los que aportan la mayor cantidad de información. En ambas pruebas, los atributos que más aumentan el *accuracy* de la clasificación corresponden a los atributos Skipgrams y Cgrams, seguidos en menor parte por Pointedness y Contextual/Temporal Imbalance. A pesar de esto, se aprecia que los atributos cercanos a elementos semánticos y psicológicos aportan de forma positiva a la clasificación en la mayoría de los casos (hay algunas excepciones como la aplicación del atributo Imagery en la Prueba 1, o el atributo Pleasantness en la Prueba 2, los cuales reducen el *accuracy* en vez de aumentarlo).

Sin embargo, se concluye que estas pruebas no bastan para descartar de forma definitiva algunos de los atributos. Esto se debe a la naturaleza de las herramientas utilizadas en la implementación de la extracción de atributos de tipo semántico y psicológico. Como se aprecia en los resultados de la extracción de atributos, algunos de ellos no tuvieron una cobertura significativa de los textos analizados. Por ejemplo, el atributo Temporal Imbalance tuvo una detección de patrones muy baja, o los atributos de la dimensión Emotional Contexts se basaban en un diccionario con una cantidad limitada de palabras. Las herramientas de análisis lingüístico y psicológico utilizadas en este trabajo, representan una gran limitante al momento de extraer los atributos y caracterizar el texto a partir de ellos. Herramientas básicas en análisis de sentimientos para el idioma Español como el diccionario de polaridad, y otras más complejas como las que permiten obtener medidas de similitud semántica (en este trabajo se utilizó Spanish JavaSimlib) o el *Dictionary of Affect in Language* en Español son relativamente nuevas o se encuentran en un estado experimental, por lo que se encuentran lejos del desempeño que muestran herramientas similares para otros idiomas como Inglés. Esto produce que los atributos más sofisticados (semánticos y psicológicos) no puedan ser explotados en su totalidad, y por ende, éstos no tienen el impacto deseado en los resultados. Para realizar un análisis más acabado de los atributos que componen este modelo y su comportamiento en el idioma Español, primero es necesario hacer pruebas contando con herramientas y recursos lingüísticos más desarrollados. Se propone que ésto componga un trabajo futuro a realizar, para posteriormente analizar la real efectividad de cada atributo.

Respecto a los resultados obtenidos en la Prueba 1, se aprecia que el modelo es capaz de permitir que una máquina diferencie texto irónico y no irónico en función del contexto en el que se está comparando (en este caso, textos irónicos versus textos objetivos). Pero, en la Prueba 2 los resultados son bajos y en la discusión se le atribuye en parte a la metodología seguida para construir el corpus. Ante esto, es necesario analizar las capacidades propias del modelo considerando que la Prueba 2 utiliza como set negativo un conjunto de textos evaluados por personas como no irónicos. ¿Qué pasaría si es el modelo el que no es capaz de capturar aquella característica que hizo que una determinada persona estableciera que cierto texto era no irónico? Tal vez, aparte de un set negativo de calidad baja, el problema se asocia a que el modelo aun no es perfecto, y requiere de atributos más cercanos a la interpretación que las personas hace de este tipo de expresiones.

Por este motivo, se propone que la investigación futura, no solo se enfoque en nuevas formas de construir un corpus para caracterizar el fenómeno o de nuevas herramientas, si no que también considere en capturar otros elementos asociados al razonamiento que hacen las personas al momento de interpretar una expresión y decidir cuando se enfrentan a un texto irónico o no irónico.

Capítulo 7

Conclusiones

En este trabajo se ha realizado una implementación propia del modelo complejo de detección de ironía propuesto por Reyes et al. (2013) [5] con la diferencia de que el idioma utilizado corresponde al Español. El modelo fue implementado para que formara parte de un clasificador para la detección de ironía en texto utilizando Machine Learning y sus técnicas asociadas. Para su desarrollo, se llevó a cabo un extenso estudio del Estado del Arte asociado al tema, junto con una interpretación de cada parte del modelo, y la utilización de una serie de herramientas tecnológicas.

Los resultados del proceso de extracción de atributos y de clasificación dejaron en evidencia una serie de falencias asociadas a algunos elementos clave de esta implementación. Uno de ellos corresponde a los recursos lingüísticos requeridos para la caracterización de cada atributo, los cuales deben estar preparados para trabajar con el idioma Español. Durante la investigación, esto representó uno de los mayores desafíos ya que gran parte de estos recursos están disponibles solo para otros idiomas como el Inglés y si llegan a estar disponibles en el idioma Español, su desempeño y fidelidad son inferiores a los desarrollados para el Inglés. La gran parte de las herramientas utilizadas fueron implementadas en base a otras investigaciones, o corresponden a los resultados de investigaciones que se encuentran dentro del Estado del Arte del área, con no mas de 2 a 5 años de antigüedad en muchos casos. Las falencias asociadas a estas herramientas muestran que, si se busca obtener resultados realmente satisfactorios en aplicaciones de Opinion Mining y Análisis de Sentimientos en Español, es necesario que las herramientas disponibles alcancen niveles iguales o superiores a los que en la actualidad exhiben las herramientas en otros idiomas.

El comportamiento del modelo fue similar al que muestra la implementación del estudio original, en cuanto a su alta dependencia respecto a los datos utilizados como contraparte negativa o no irónica. Pero, hay que considerar que el proceso para generar el set de datos irónicos fue refinado en comparación al utilizado en la implementación original, ya que los datos fueron evaluados utilizando el discernimiento de las personas, las cuales son consideradas por muchos autores como buenos identificadores de ironía. Sin embargo, el cálculo del estadístico *kappa de Cohen* dejo en evidencia que la evaluación de este tipo de información no es tan sencilla incluso para un humano, si no se cuenta con la información o el contexto

adecuado. La metodología seguida para refinar este set de datos no fue del todo exitosa, considerando que a pesar de que una persona adulta y sana puede ser capaz de detectar ironía, ésto también depende del contexto y la naturaleza del texto en cuestión. Es decir, la presencia de un evaluador humano no garantiza que sea posible etiquetar textos irónicos o no irónicos correctamente en todos los casos evaluados. Para este tipo de evaluaciones debería aplicarse una método que cuente con una mayor cantidad de participantes para generar un set con un mayor grado de concordancias entre sus evaluadores.

En este trabajo en particular, los resultados obtenidos en la Prueba 2 (donde se obtiene el nivel de *accuracy* más bajo) se pueden analizar desde dos perspectivas. Una se vincula a la naturaleza del origen de los datos, los cuales contienen una intención comunicativa previa a su evaluación por parte de personas, la cual es transmitir un mensaje con ironía (o relacionados con algo irónico) en la mayoría de los casos extraídos. Esta naturaleza produce que la evaluación del set negativo de datos utilizados en la Prueba 2, sea menos certera que la evaluación de los casos positivos. O sea, se genera un set positivo de calidad aceptable, pero un set negativo que no tiene una gran diferencia respecto al set positivo, por lo que el resultado de la clasificación se ve afectado negativamente. Por otro lado, existe la posibilidad de que hayan elementos o detalles que las personas toman en consideración al momento de decidir si una expresión corresponde a algo irónico o no, que el modelo es incapaz de capturar. Entonces, el hecho de usar de un set mucho más refinado también impacta en los resultados al no contar con un modelo que capturase estas características utilizadas para refinar los datos.

En relación a los atributos semánticos del modelo, si bien es cierto que su desempeño se vio afectado debido a la baja fidelidad de las herramientas tecnológicas asociadas, el modelo original mostró que los niveles de información de algunos eran bajos. En primera instancia no se descarta ningún atributo ya que el trabajo fue desarrollado en un contexto distinto, con datos de origen distinto y un idioma distinto por lo que se optó por no asumir que el comportamiento del modelo sería similar. Esto deja en evidencia que es necesario realizar una revisión a la efectividad de los atributos del modelo y utilizar medidas de selección de atributos según los niveles de información que aportan al modelo. Además, se debe considerar que la ironía no es un fenómeno que pueda caracterizarse de forma universal para todo el mundo. Existen elementos ligados a la idiosincrasia de cada sector del mundo que pueden influir en cómo se expresa e interpreta la ironía, por lo que tal vez sea recomendable buscar soluciones a nivel local que sean más efectivas que una única solución universal, la cual es compleja de establecer. También sería útil preparar los datos para otros métodos de clasificación y analizar el comportamiento de los atributos en función de ellos, considerando que existen otros algoritmos de clasificación que podrían mostrar mejores resultados. Ésto no se realizó en este trabajo ya que no se encontraba dentro de sus alcances, pero se propone como trabajo futuro.

Este trabajo constituye uno de los primeros acercamientos a la construcción de una herramienta que permita la detección automática de la presencia de ironía en texto en idioma Español (si es que no es la primera en ser desarrollada). Desde la perspectiva del área de Opinion Mining, esta área es bastante nueva y la investigación asociada al tema es escasa, sobretodo en Español donde es casi inexistente. Esto se debe a que el área de Opinion Mining en la Web en si es un área bastante nueva y el caso particular de ironía es considerado como

un caso complejo, muchas veces descartado por su alto nivel de complejidad. A pesar de ésto, el trabajo realizado y sus resultados conforman un buen punto de inicio para el desarrollo de la investigación en torno a este fenómeno. Los resultados, a pesar de no ser completamente satisfactorios, muestran en relación a la Hipótesis de Investigación que es posible detectar la presencia de ironía a ciertos niveles y bajo ciertas condiciones, ya que en este caso el modelo no es perfecto. A pesar de ésto, se considera que se cumple la Hipótesis de Investigación ya que es posible detectar la presencia de ironía en idioma Español, aunque esta detección no sea una solución absoluta. Ante esto, se hace hincapié en que es necesario extender la investigación e incluir otros elementos que caractericen el fenómeno y no solo a nivel general, sino que también a nivel local respecto al lenguaje y la cultura en la que se está inmerso.

Respecto a la utilidad del desarrollo de esta área, es posible señalar que más allá de su objetivo original (el cual es potenciar el desempeño del Análisis de Sentimientos) la detección de ironía podría transformarse en una función útil en si misma debido a la alta presencia de ironía en los distintos medios sociales donde las personas expresan su opinión constantemente como Twitter. Aunque esto no se encuentra medido en números, basta con navegar a través de Twitter para saber que muchas personas utilizan este recurso para expresar sus opiniones y críticas. Para una campaña política o para un estudio de mercado, puede ser interesante conocer el nivel de ironía contenido en las opiniones respecto a ciertos productos, candidatos o medidas políticas, etc. Sin embargo, aun no se puede pensar en generar soluciones comerciales ya que la detección está lejos de ser perfecta. Incluso, empresas inmersas en el rubro de análisis de redes sociales consideran que la detección automática de ironía es extremadamente compleja o imposible sin la participación de un humano. A pesar de esto, una función de éste tipo puede agregar valor al perfeccionar los análisis de reputación que se realizan utilizando el contenido Web generado a partir de redes sociales. Como cualquier otra aplicación basada en Análisis de Sentimientos, los sistemas de análisis de reputación pueden verse afectados por el uso extensivo de ironía en el contenido que utilizan para generar reportes a sus clientes (empresas, partidos políticos, etc.) y por esto, ya existen empresas en el rubro que tienen en consideración este fenómeno y su impacto[1].

Por último, se puede agregar que esta investigación agrega valor al proyecto *OpinionZoom* al aportar el conocimiento necesario respecto a la actualidad en el estudio del fenómeno de la ironía, y representa un primer paso para el largo camino de trabajo que éste requiere. Los resultados no son perfectos y no son suficientes para que el módulo de detección forme parte de los servicios que OpinionZoom ofrece, sin embargo, el conocimiento generado a partir de esta investigación respecto a cada elemento utilizado en su realización (herramientas tecnológicas varias, diccionarios de análisis de sentimientos, cálculo de medidas de similitud semántica) representan un aporte que podrá ser reutilizado en investigaciones futuras y/o nuevas funciones para el servicio que OpinionZoom ofrece. En síntesis, este trabajo es valioso para el conocimiento al interior de OpinionZoom y aumenta las posibilidades para futuras investigaciones al haber entrado en *terrenos desconocidos* utilizando nuevas herramientas que forman parte del Estado del Arte de Opinion Mining.

[1] http://www.brandrain.com/blog/el-camino-hacia-la-deteccion-de-la-ironia-reto-para-el-analisis-de-la-reputacion/

7.1 Trabajo Futuro

A partir de las conclusiones generadas por los resultados de este trabajo, se puede notar que existen puntos que deben ser abarcados como trabajo futuro, para poder complementar la investigación asociada:

- Es necesario disponer de recursos lingüísticos semánticos que tengan una mayor fidelidad, con resultados sobresalientes a nivel experimental. Estos son escasos para el idioma Español por lo que sería útil que el desarrollo siga evolucionando.
- En base al punto anterior, sería interesante probar el desempeño del modelo actual, una vez que se cuente con recursos lingüísticos más desarrollados, para que la extracción de atributos sea más completa.
- Complementar la construcción de un set de datos irónico con una evaluación más completa, con el objetivo de establecer un corpus construido a modo de *estándar de oro* para que pueda ser utilizado de forma amplia en el campo de investigación.
- Ligado al punto anterior, sería interesante disponer de un set de datos de mayor extensión, para la realización de las pruebas de clasificación. Si se pretende realizar un set evaluado, se podría utilizar un enfoque semi-supervisado para esta construcción.
- Realizar una selección de atributos para el modelo a través de las metodologías estadísticas establecidas, con el objetivo de evaluar su desempeño y mejorar sus características.
- Analizar la posibilidad de agregar nuevos atributos al modelo para cubrir otros elementos que caractericen el fenómeno de ironía. Como se ha mostrado, la definición es amplia por lo que es un trabajo que da para bastante investigación.
- Pensando tanto en la construcción del corpus como en los atributos que captura el modelo, resultaría interesante analizar elementos más cercanos al medio sociocultural local en el que se está inmerso, y ver su influencia en el fenómeno de la ironía para dar con un modelo más acotado y certero.
- Por último, se debe probar el impacto de las predicciones de este modelo, sobre el desempeño de un clasificador de polaridad. Esto es uno de los fundamentos principales del interés en el estudio del fenómeno de ironía por lo que se debería aplicar una vez que se obtengan mejores resultados.

Referencias

[1] R. T. S. Project, "Internet live stats." `http://www.internetlivestats.com/internet-users/`, 2014. visto en 25/07/2014.

[2] Q. Wang, K. Y. Goh, and X. Lu, "How does user-generated content influence consumers' new product exploration? an empirical analysis of consumer search and choice behaviors," *An Empirical Analysis of Consumer Search and Choice Behaviors (August 5, 2013)*, 2013.

[3] J. D. Velásquez and L. C. Jain, *Advanced techniques in web intelligence*, vol. 1. Springer, 2010.

[4] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and trends in information retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008.

[5] A. Reyes, P. Rosso, and T. Veale, "A multidimensional approach for detecting irony in twitter," *Language Resources and Evaluation*, vol. 47, no. 1, pp. 239–268, 2013.

[6] B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[7] B. Seerat and F. Azam, "Opinion mining: Issues and challenges (a survey)," *International Journal of Computer Applications (0975–8887) Volume*, 2012.

[8] T. Berners-Lee and R. Cailliau, "Worldwideweb: Proposal for a hypertext project," *Retrieved on February*, vol. 26, p. 2008, 1990.

[9] R. Kaleelazhicathu, "History of internet pricing,"

[10] D. Best, "Web 2.0: Next big thing or next big internet bubble," *Technische Universiteit Eindhoven*, 2006.

[11] S. Aghaei, M. A. Nematbakhsh, and H. K. Farsani, "Evolution of the world wide web: From web 1.0 to web 4.0," *International Journal of Web & Semantic Technology*, vol. 3, no. 1, pp. 1–10, 2012.

[12] E. Constantinides and S. J. Fountain, "Web 2.0: Conceptual foundations and marketing issues," *Journal of Direct, Data and Digital Marketing Practice*, vol. 9, no. 3, pp. 231–244, 2008.

[13] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "The kdd process for extracting useful knowledge from volumes of data," *Commun. ACM*, vol. 39, pp. 27–34, Nov. 1996.

[14] O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook*, vol. 14. Springer, 2010.

[15] J. Han and M. Kamber, *Data Mining: Concepts and Techniques 3rd Ed.* Morgan kaufmann, 2012.

[16] R. Cooley, B. Mobasher, and J. Srivastava, "Web mining: Information and pattern discovery on the world wide web," in *Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on*, pp. 558–567, IEEE, 1997.

[17] U. M. Patil and J. Patil, "Web data mining trends and techniques," in *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*, pp. 961–965, ACM, 2012.

[18] H. Chen and M. Chau, "Web mining: Machine learning for web applications," *Annual review of information science and technology*, vol. 38, pp. 289–330, 2004.

[19] E. Marrese Taylor, "Diseño e implementación de una aplicación de web opinion mining para identificar preferencias de usuarios sobre productos turísticos de la x región de los lagos," 2013.

[20] P. A. Tapia Caro, "Diseño e implementación de un sistema para la clasificación de tweets según su polaridad," 2014.

[21] G. A. Martínez Azocar, "Mejoramiento de una metodología para la identificación de website keyobject mediante la aplicación de tecnologías eye tracking, análisis de dilatación pupilar y algoritmos de web mining," 2013.

[22] D. Maynard, K. Bontcheva, and D. Rout, "Challenges in developing opinion mining tools for social media," *Proceedings of@ NLP can u tag# usergeneratedcontent*, 2012.

[23] E. Filatova, "Irony and sarcasm: Corpus generation and analysis using crowdsourcing.," in *LREC*, pp. 392–398, 2012.

[24] E. Alpaydin, *Introduction to Machine Learning*. The MIT Press, 2010.

[25] A. Hotho, A. Nürnberger, and G. Paaß, "A brief survey of text mining.," in *Ldv Forum*, vol. 20, pp. 19–62, 2005.

[26] G. G. Chowdhury, "Natural language processing," *Annual review of information science and technology*, vol. 37, no. 1, pp. 51–89, 2003.

[27] E. D. Liddy, "Natural language processing," 2001.

[28] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*, vol. 1. Cambridge university press Cambridge, 2008.

[29] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python*. O'Reilly Media, Inc., 2009.

[30] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*. MIT press, 1999.

[31] D. L. Olson and D. Delen, *Advanced data mining techniques*. Springer Science & Business Media, 2008.

[32] A. Reyes and P. Rosso, "Building corpora for figurative language processing: The case of irony detection," in *Proceedings of the 4th International Workshop on Corpora for Research on Emotion Sentiment & Social Signals (in conjunction with LREC 2012)*, pp. 94–98, 2012.

[33] B. A. Nardi, D. J. Schiano, and M. Gumbrecht, "Blogging as social activity, or, would you let 900 million people read your diary?," in *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pp. 222–231, ACM, 2004.

[34] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: understanding microblogging usage and communities," in *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pp. 56–65, ACM, 2007.

[35] Twitter, "About twitter." `http://about.twitter.com/company`, 2014. visto en 05/12/2014.

[36] Twitter, "Twitter api overview." `http://dev.twitter.com/overview`, 2014. visto en 05/12/2014.

[37] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?," in *Proceedings of the 19th international conference on World wide web*, pp. 591–600, ACM, 2010.

[38] B. A. Huberman, D. M. Romero, and F. Wu, "Social networks that matter: Twitter under the microscope," *arXiv preprint arXiv:0812.1045*, 2008.

[39] R. W. Gibbs and H. L. Colston, *Irony in language and thought: A cognitive science reader*. Psychology Press, 2007.

[40] H. H. Clark and R. J. Gerrig, "On the pretense theory of irony.," 1984.

[41] D. Wilson and D. Sperber, "On verbal irony," *Lingua*, vol. 87, no. 1, pp. 53–76, 1992.

[42] J. Lucariello, "Situational irony: A concept of events gone awry.," *Journal of Experimental Psychology: General*, vol. 123, no. 2, p. 129, 1994.

[43] S. Dews, J. Kaplan, and E. Winner, "Why not say it directly? the social functions of irony," *Discourse processes*, vol. 19, no. 3, pp. 347–367, 1995.

[44] R. W. Gibbs, "Irony in talk among friends," *Metaphor and symbol*, vol. 15, no. 1-2, pp. 5–27, 2000.

[45] P. Carvalho, L. Sarmento, M. J. Silva, and E. de Oliveira, "Clues for detecting irony in user-generated contents: oh...!! it's so easy;-)," in *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pp. 53–56, ACM, 2009.

[46] D. Davidov, O. Tsur, and A. Rappoport, "Semi-supervised recognition of sarcastic sentences in twitter and amazon," in *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pp. 107–116, Association for Computational Linguistics, 2010.

[47] A. Reyes and P. Rosso, "Mining subjective knowledge from customer reviews: a specific case of irony detection," in *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, pp. 118–124, Association for Computational Linguistics, 2011.

[48] R. González-Ibáñez, S. Muresan, and N. Wacholder, "Identifying sarcasm in twitter: a closer look," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pp. 581–586, Association for Computational Linguistics, 2011.

[49] C. de Ayuda de Twitter, "¿qué son las etiquetas?." `https://support.twitter.com/articles/247830-que-son-las-etiquetas-simbolos#`, 2014. visto en 15/10/2014.

[50] J. L. Fleiss, B. Levin, and M. C. Paik, *Statistical methods for rates and proportions*. John Wiley & Sons, 2013.

[51] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining.," in *LREC*, vol. 10, pp. 1320–1326, 2010.

[52] E. Stamatatos *et al.*, "Ensemble-based author identification using character n-grams," in *Proceedings of the 3rd International Workshop on Text-based Information Retrieval*, pp. 41–46, Citeseer, 2006.

[53] I. Kanaris, K. Kanaris, I. Houvardas, and E. Stamatatos, "Words versus character n-grams for anti-spam filtering," *International Journal on Artificial Intelligence Tools*, vol. 16, no. 06, pp. 1047–1067, 2007.

[54] V. Kešelj, F. Peng, N. Cercone, and C. Thomas, "N-gram-based author profiles for authorship attribution," in *Proceedings of the conference pacific association for computational linguistics, PACLING*, vol. 3, pp. 255–264, 2003.

[55] D. Guthrie, B. Allison, W. Liu, L. Guthrie, and Y. Wilks, "A closer look at skip-gram modelling," in *Proceedings of the 5th international Conference on Language Resources and Evaluation (LREC-2006)*, pp. 1–4, 2006.

[56] C. Whissell, "Using the revised dictionary of affect in language to quantify the emotional undertones of samples of natural language 1, 2," *Psychological reports*, vol. 105, no. 2, pp. 509–521, 2009.

[57] M. Lutz, *Learning python, 4th Edition*. O'Reilly Media, Inc., 2009.

[58] S. M. Tahaghoghi and H. E. Williams, *Learning MySQL*. O'Reilly Media, Inc., 2006.

[59] D. J. Eck, *Introduction to programming using Java*. David J. Eck, 2006.

[60] G. Hackeling, *Mastering Machine Learning with scikit-learn*. Packt Publishing Ltd, 2014.

[61] I. Idris, *NumPy Beginner's Guide*. Packt Publishing Ltd, 2013.

[62] I. Lozano, A. Trilla, and F. Alías, "Spanish javasimlib: a tool to compute the semantic similarity between words in spanish," 2012.

[63] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *arXiv preprint cmp-lg/9709008*, 1997.

[64] F. L. Cruz, J. A. Troyano, B. Pontes, and F. J. Ortega, "Building layered, multilingual sentiment lexicons at synset and lemma levels," *Expert Systems with Applications*, vol. 41, no. 13, pp. 5984–5994, 2014.

[65] F. L. Cruz, J. A. Troyano, B. Pontes, and F. J. Ortega, "Ml-senticon: Un lexicón multilingüe de polaridades semánticas a nivel de lemas," *Procesamiento del Lenguaje Natural*, vol. 53, pp. 113–120, 2014.

[66] J. Brooke, M. Tofiloski, and M. Taboada, "Cross-linguistic sentiment analysis: From english to spanish.," in *RANLP*, pp. 50–54, 2009.

[67] M. G. D. Rıos and A. Gravano, "Spanish dal: A spanish dictionary of affect in language," *WASSA 2013*, p. 21, 2013.

[68] E. Filippova and J. W. Astington, "Further development in social reasoning revealed in discourse irony understanding," *Child development*, vol. 79, no. 1, pp. 126–138, 2008.

[69] S. Shamay-Tsoory, R. Tomer, and J. Aharon-Peretz, "The neuroanatomical basis of understanding sarcasm and its relationship to social cognition," *Neuropsychology*, vol. 19, no. 3, pp. 288–300, 2005.

Apéndices

A Utilidades de Implementación

A .1 Pre-procesamiento

- **Expresión Regular para Tokenización:**

```
[([\(])([HhLlKk8])([\)])|]
[(\^|\-|\*|\¬|[oOuUT0])(|o|\.|\_)(\^|\-|\*|\¬|[oOuUT0])|
(:|=|;|8|B)(|o|O|-|=|>)([pP]|[doOcCsSL\$|\/]|[\(\[]|[D\)\]]|[/\|/]|[\\\\])|
([pP]|[doOcCsSL\$|\/]|[\(\[]|[D\)\]]|[/\|/]|[\\\\])(|o|O|-|=|>)(:|=|;|8|B)|
([\<])([3]|\/3|[\\\3])|
(?x)([A-Z]\.)+|
\w+(-\w+)*|
([xX])([dD])|
\$?\d+(\.\d+)?%?|
\.+| \?+| \¿+| \¡+| \!+|
[][,;"'¿?():-_`¡!]
```

Figura 7.1: Expresión Regular para Tokenización
Fuente: Elaboración propia.

- **Lista de Emoticonos para Pre-procesamiento:**

```
:), :-), :o, :-o, :O, :-O, =o, =O, :(, :-(, =(, :c, =c, :-c, :C, =C, :-C, :=),
:P, :-P, :p, :-p, :D, =D, :-D, :d, :-d, :$, =$, :-$, :/, =/, :-/, :\, =\, :-\,
:], =], :-], :[, =[, :-[, :L, =L, :-L, :s, =s, :-s, :S, =S, :-S, :|, =|, :-|,
;), ;-), =), ;o, ;-o, ;O, ;-O, ;(, ;-(, ;c, ;-c, ;C, ;-C,
;=), ;=), ;P, ;-P, ;p, ;-p, ;D, ;-D, ;d, ;-d, ;$, ;-$, ;/, ;-/, ;\, ;-\,
;], ;-], ;[, ;-[, ;L, ;-L, ;s, ;-s, ;S, ;-S, ;|, ;-|,
8], 8-], 8), 8-), B), B-), 8[, 8-[, 8(, 8-(, B(, B-(,
(:, (-:, o:, o-:, O:, O-:, o=, O= ):, )-:, )=, c:, c=, c-:, C:, C=, C-:,
(=:, D:, D=, D-:, $:, $=, $-:, /:, /=, /-:, \:, \=, \-:,
[:, [=, [-:, ]:, ]=, ]-:, s:, s=, s-:, S:, S=, |:, |=, |-:,
(;, (-;, (=, o;, o-;, O;, O-;, );, )-;, c;, c-;, C;, C-;,
(=;, D;, D-;, $;, $-;, /;, /-;, \;, \-;,
[;, [-;, ];, ]-;, s;, s-;, S;, S;, |;, |-;,
[8, [-8, (8, (-8, ]8, ]-8, )8, )-8,
^_^, -_-, *_*, o_o, O_O, O_o, o_O, 0_0, ¬_¬, u_u, U_U, T_T,
^.^, -.-, *.*, o.o, O.O, O.o, o.O, 0.0, ¬.¬, u.u, U.U, T.T,
^o^, *o*,
xD, XD, Xd, xd, 1313, lol, LOL, ¬¬,
(H), (h), (L), (l), (K), (k), (Y), (y), (8), <3, </3, <\3
```

Figura 7.2: Lista de Emoticonos para Pre-procesamiento
Fuente: Elaboración propia.

A .2 Implementación Dimensión Signatures

- **Lista de adverbios para atributo Counterfact:**

  aproximadamente, alrededor, casi, cerca, entonces,
  incluso, más, menos, no, sólo, simplemente,
  sencillamente, todavía, aún, virtualmente,
  mas, solo, todavia, aun, aunque, ahora, pero,
  con todo, de allí, no obstante, sin embargo,
  de alli, a pesar de, más o menos, por lo tanto, punto menos que,
  mas o menos

- **Lista de adverbios para atributo Temporal Compression:**

  abruptamente, improvisadamente, después, luego, ahora,
  inesperadamente, recientemente, recién, últimamente, pronto,
  ayer, desde, repentino, súbito, rápido, precipitado, repentinamente,
  mañana, despues, recien, ultimamente, subito, rapido,
  más tarde, sin avisar, en breve, desde entonces, de repente,
  cuando sea, cuando quiera, mas tarde, de la nada, dentro de poco