PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

Implementación de una herramienta de automatización para la simplificación de textos en español

Tesis para optar el Título de **Ingeniero Informático**, que presenta el bachiller:

Luis Fernando Muroya Tokushima

ASESOR: Fernando Alva Manchego

Lima, junio del 2015

Resumen

El acceso a la información es un derecho reconocido en el artículo 19 de la Declaración Universal de los Derechos Humanos e implica, entre muchas cosas, que la información disponible a través de los medios escritos sea comprensible para todos. Sin embargo, en pleno siglo XXI, todavía existe una brecha insalvable entre la idealidad y la realidad: muchas personas no pueden entender los textos porque son muy complejos para ellos. Los motivos son muy diversos: una educación deficiente, alguna forma de discapacidad, factores exógenos, etc.

Para que estas personas gocen de un verdadero acceso a la información, es necesario que la misma sea transformada y reescrita de forma comprensible para ellas, proceso conocido como simplificación textual. En vista que llevarla a cabo manualmente resulta costoso, en los últimos años se ha vuelto de interés automatizar este proceso. En el caso del español, un primer avance se logró con el proyecto Simplext (2009), durante el cual fue construida una herramienta de simplificación léxico-sintáctica con un enfoque primordialmente basado en reglas.

En este trabajo se desarrolló una herramienta de simplificación que opera a nivel sintáctico y utiliza clasificadores como elementos de decisión para escoger las operaciones de simplificación sintáctica que deben ser aplicadas sobre un texto. Para construir esta herramienta, primero se anotó el corpus paralelo recopilado durante el proyecto Simplext con las operaciones de simplificación que se llevaron a cabo oración por oración. Asimismo, se entrenó clasificadores binarios y se llevó a cabo una experimentación numérica con la finalidad de definir cuál era el más adecuado para cada operación. Finalmente, se implementó una interfaz web que permite al usuario ingresar un texto y visualizar su versión simplificada.

Luego de comparar el índice de perspicuidad (métrica de complejidad textual) de textos antes y después de pasar por la herramienta, se pudo concluir que la misma sí es capaz de simplificar textos, aunque no de una forma tan eficiente como un ser humano. Esta diferencia era esperada debido a que no se incluyó un módulo de simplificación léxica y porque un agente humano tiene experiencia y conocimiento que una máquina carece. En un trabajo futuro podría integrarse dicho módulo para mejorar el funcionamiento de la herramienta.

Tabla de contenido

CAPÍTULO 1

"Everyone has the right to freedom of opinion and expression; this right includes freedom to hold opinions without interference and to seek, receive and impart information and ideas through any media and regardless of frontiers" (The Universal Declaration of Human Rights, 1948).

1 Problemática

El acceso a la información es un derecho universal, tal como lo reconoce la ONU en el artículo 19 de la Declaración Universal de Derechos Humanos, según el cual todos tienen derecho a buscar, recibir e impartir información por cualquier medio y sin tener en cuenta fronteras. En el siglo 21, este intercambio universal de información se ha hecho posible gracias al Internet y se ha vuelto parte esencial de la vida diaria. Sin embargo, existe un gran número de personas que no puede ejercer este derecho a plenitud puesto que no son capaces de comprender la información a la que pueden acceder por medios escritos (Drndarević y Saggion, 2012).

En un estudio llevado a cabo por la ONU en el 2006 acerca de la accesibilidad de la información en Internet, de una muestra de 100 sitios web importantes, sólo un 3% pasó la prueba de accesibilidad básica (Drndarević y Saggion, 2012). De acuerdo a (W3C, 2012), la accesibilidad web involucra que cualquier persona pueda percibir, comprender, navegar, contribuir e interactuar con la página web.

Con el estudio anteriormente mencionado se observó que mucha de la información disponible en línea es percibida como incomprensible por muchas personas; en particular, individuos con diferentes discapacidades cognitivas[1] (Drndarevic y Saggion, 2012), quienes representan al menos el 5% de la población mundial según estimaciones de la Asociación Fácil Lectura (*apud* Saggion *et. al.*, 2012). Es por ello que en el año 2007, la ONU reconoció el derecho a un acceso

[1] Las discapacidades cognitivas son desórdenes neurológicos que afectan las capacidades del cerebro para recibir, procesar, almacenar y responder a información. Un típico ejemplo es la dislexia (NCLD, 2014). En el contexto descrito por el autor de la cita, sin embargo, se puede incluir también a personas con retraso mental.

fácil a contenido digital para las personas con discapacidades intelectuales o con dificultades de lenguaje (Saggion *et. al.*, 2012).

Es importante tener en cuenta que una persona sin discapacidad puede tener un acceso limitado a la información por poseer un nivel bajo de comprensión lectora. Esta situación es señalada por (Allington, 2012) al indicar que “no se puede aprender de libros que no se pueden leer”.

En un contexto local, el bajo nivel de comprensión lectora en el Perú desde la etapa escolar es una realidad conocida. En la Evaluación Censal de Estudiantes del 2013, un estudio anual realizado por el Ministerio de Educación, se observó que un 15.8% de alumnos de segundo grado de primaria solo fue capaz de responder preguntas muy básicas de los textos leídos; un 51.3% de la muestra no fue capaz de realizar inferencias a partir de lo leído y solo el 32.9% restante fue capaz de comprender el texto y responder las preguntas más complicadas (MINEDU, 2013).

Esta situación también se observó en el examen PISA del 2012, una evaluación que se lleva a cabo cada 3 años y cuyo objetivo es examinar el estado del sistema educativo de varios países por medio de la evaluación de las habilidades y competencias de estudiantes de diferentes grados de instrucción. En las últimas versiones del examen, se ha prestado gran atención a las habilidades de comprensión lectora, pues se trata de una habilidad transversal necesaria para el aprendizaje. En el marco de este criterio, Perú ocupó el último lugar de la lista (puesto 65 de 65); es más, el puntaje que obtuvo se encontró por debajo del promedio mundial (OECD, 2012).

En vista de lo anterior, cabe preguntar, ¿cuál es un factor que dificulta a los grupos anteriormente descritos un acceso irrestricto a información? Sin duda, existe una gran cantidad de respuestas a esta pregunta; sin embargo, en todos los casos se reconocerá que el alto grado de complejidad que tienen muchos textos es un factor crucial en el asunto (Drndarević y Saggion, 2012).

Entonces, ¿qué es lo que hace complejo un texto? Parece una pregunta sencilla; sin embargo, es importante reconocer que existe una multitud de aristas por dónde abordarla ya que se trata de una calificación subjetiva que depende de las habilidades y experiencia del lector (Saggion y Bott, 2011). No obstante, se reconoce que la estructura interna, coherencia, unidad y vocabulario son factores muy importantes en el proceso de calificar un texto como apto para un público específico o no; es decir, si es muy complejo o no (Fisher y Frey, 2012).

Por lo tanto, para mejorar el acceso a la información que tienen los grupos previamente descritos (personas con discapacidad cognitiva y con bajo nivel de comprensión lectora) es necesario que los textos sean menos complejos. Para tal fin, existen diferentes metodologías para transformar un texto en otro más sencillo y fácil de comprender. Ellas suelen atacar los factores que vuelven complejo a un texto y que fueron mencionados anteriormente. Sin embargo, requieren de un fuerte consumo de recursos humanos y económicos, por lo que resulta poco atractiva la inversión necesaria para llevarlas a la práctica (Drndarević y Saggion, 2012).

Por lo anteriormente expuesto, resulta importante la automatización del proceso de simplificación de textos; es decir, la creación de un sistema que reciba un texto y devuelva como resultado la versión simplificada del mismo. Ya hay avances en este proceso para el inglés, portugués, francés, italiano y danés; sin embargo, no ha sido muy desarrollado para el español. Es por ello que resulta de gran interés automatizar la simplificación de textos en dicho idioma (Saggion y Bott, 2014)

Estudios llevados a cabo previamente han demostrado que para conseguir dicho objetivo, se requiere principalmente de dos módulos: uno sintáctico y otro léxico. El primero, que es de gran importancia para la simplificación textual, ha sido trabajado en base a reglas estáticas en una investigación previa y, según trabajos recientes de (Saggion *et. al.*, 2012), tiene una eficiencia limitada pues la aplicación de estas reglas genera ambigüedades con mucha frecuencia.

Un módulo integrado de simplificación sintáctica de español basado en métodos probabilísticos aún no ha sido propuesto; no obstante, sí lo ha sido para sistemas análogos en inglés (Woodsend y Lapata, 2011-a; Woodsend y Lapata, 2011-b) y portugués (Aluísio y Gasperín, 2010). Los resultados obtenidos en ellos han demostrado que los módulos de simplificación sintáctica resultantes son más eficientes que sus contrapartes basadas en reglas. Por ello, es de esperar que aplicar dicho enfoque en la simplificación sintáctica de textos en español tenga un resultado positivo.

En resumen, el proyecto consiste en la creación de una herramienta para la simplificación sintáctica de textos en español; es decir, deberá disminuir la complejidad (medidas con métricas lingüísticas) de los mismos. Esta será construida utilizando técnicas de aprendizaje de máquina porque se espera que tenga un mejor desempeño que una construida con reglas estáticas.

2 Marco conceptual

En esta sección serán presentados los conceptos identificados como claves para la comprensión del presente proyecto. En particular, se abordarán temas relacionados al lenguaje, niveles de análisis del lenguaje, procesamiento de lenguaje natural, simplificación textual y aprendizaje de máquina.

2.1 El lenguaje y sus niveles de análisis

"-(José Arcadio Buendía) Es el diamante más grande del mundo.
-No –corrigió el gitano-. Es hielo".
(Gabriel García Márquez, Cien Años de Soledad, 1967)

La Real Academia Española define el lenguaje como un conjunto de sonidos articulados con que el hombre manifiesta lo que piensa o siente. En una segunda acepción de la palabra, lo define como lengua; es decir, un sistema de comunicación verbal y casi siempre escrito propio de una comunidad humana (RAE, 2013).

Cómo adquiere el ser humano una habilidad tan maravillosa y que le ha permitido distinguirse del resto de seres vivos sobre el planeta ha sido motivo de debate para los lingüistas, quienes están divididos en dos corrientes: los racionalistas, que apoyan la teoría de Chomsky y los empiristas. Los primeros sostienen que el ser humano tiene un conocimiento innato del lenguaje y su estructura mientras que los últimos apoyan la idea que el ser humano nace con un conjunto de habilidades (reconocimiento de patrones, asociaciones, generalizaciones) que le permite aprender la estructura del lenguaje (Manning, 1999). El aprendizaje del lenguaje por el ser humano es un amplio campo de estudio y sus detalles escapan del alcance de esta revisión.

A pesar de las diferentes opiniones de cómo el ser humano adquiere el conocimiento de un lenguaje, sí existe un consenso sobre cómo está estructurado. Los expertos coinciden que este conocimiento se da a través de diferentes niveles que se traslapan entre sí, complementándose. Según (Jurafsky y Martin, 2007) son los siguientes:

- **Morfológico:** es el conocimiento de la estructura de las palabras así como de las partes que le dotan su significado y de la categoría a la que pertenece. Por ejemplo, permite identificar qué parte de la palabra define el

género y el número de la misma, contracciones, etc. En el caso del ejemplo tomado de la famosa obra de Gabriel García Márquez y que se encuentra al principio de esta sección, el gitano le dice a un despistado José Arcadio Buendía: "*es hielo*". Un análisis morfológico de la palabra *hielo* revela que es un sustantivo propio, masculino y singular.

- **Sintáctico:** se trata del conocimiento de las relaciones estructurales entre palabras; es decir, el conocimiento necesario para ordenar y agrupar las palabras en una oración para que el mensaje pueda ser entendido. Así, en el ejemplo, José Arcadio Buendía afirma "*Es el diamante más grande del mundo*" y no dice "*Es más diamante grande del mundo*" porque su interlocutor no lo comprendería.
- **Léxico y Semántico:** es el conocimiento del significado de las palabras como unidades (léxico) y de la estructura que forman al agruparse (semántico). En el ejemplo, el gitano reconoce que su interlocutor está afirmando que lo que ve es un diamante.
- **Pragmático:** se trata del conocimiento de las relaciones entre el significado y las intenciones del hablante. Así, en el diálogo, el gitano identifica que José Arcadio Buendía está haciendo una aseveración y no está formulando una pregunta ni una petición.
- **Discursivo:** es el conocimiento acerca de unidades lingüísticas que van más allá de una oración. Se trata de una tarea de resolución de referencias. Así, en el ejemplo se reconoce que al decir "*es hielo*" se está haciendo referencia al mismo objeto al que se refiere José Arcadio.
- **Fonético:** es aplicado al lenguaje hablado. Hace referencia al conocimiento de los sonidos lingüísticos. Por ejemplo, en el diálogo, cada interlocutor es capaz de comprender los sonidos que emite el otro y reconocer en ellos las palabras.

2.2 El campo del Procesamiento de Lenguaje Natural

(Liddy, 2001) define formalmente el Procesamiento de Lenguaje Natural como un conjunto de técnicas computacionales para analizar y representar textos que ocurren naturalmente en uno o más niveles de análisis lingüístico con el propósito de proveer a un conjunto de tareas o aplicaciones la capacidad de procesar un lenguaje cercano al humano. Dicho de otra forma, su objetivo es conseguir que las computadoras lleven a cabo tareas útiles que se les pueda indicar usando lenguaje natural; es decir, uno muy próximo al humano (Jurafsky y Martin, 2007).

2.3 Simplificación textual

La simplificación automática de textos, como indica su nombre, consiste en la transformación de un texto en uno equivalente con menor complejidad de vocabulario y estructura a través de un conjunto pequeño de operaciones (Saggion y Bott, 2011).

2.3.1 La complejidad de los textos

Un componente importante de la definición anterior lo constituye la palabra complejidad. La complejidad, según la Real Academia Española es la cualidad de algo complejo; mientras que complejo es tratado como sinónimo de complicado. Es decir, la complejidad es un indicador de cuán complicado resulta leer y comprender un texto. ¿Qué características determinan si un texto es complicado o simple? Según (CCSS, 2013), la complejidad textual está dada por tres componentes interrelacionados:

- **El componente cualitativo:** está conformado por todos los aspectos de la complejidad que sólo pueden ser medidos por un lector humano. Verbigracia, significado, estructura, claridad, convencionalidad, etc.
- **El componente cuantitativo:** está conformado por todos los aspectos de la complejidad que son difíciles (o imposibles) de medir por un lector humano; sobretodo, para textos muy extensos. Ejemplos de estas medidas son: tamaño y frecuencia de palabras, tamaño de texto, cohesión, etc.

Una medida muy usada para cuantificar la complejidad de un texto en español es el índice de Flesch-Szigriszt o "Fórmula de Perspicuidad" (Barrio-Cantalejo *et. al.,* 2008), cuya fórmula es:

$$INFZ = 206.835 - 62.3 * \frac{silabas}{palabra} - \frac{palabra}{frase} \dots (\boldsymbol{Ec.\,1.1})$$

Se trata de una métrica que define siete intervalos que se muestran en la tabla 1.1.

Tabla 1.1. Evaluación de un texto según métrica de Flesch- Szigriszt.

Puntos	Grado
0-30	Muy difícil
30-40	Difícil
40-50	Algo difícil
50-70	Normal
70-80	Algo fácil
80-90	Fácil
90-100	Muy fácil

Fuente: adaptado de (Barrio-Cantalejo et. al., 2008)

- **El componente relacionado con el lector y con la tarea:** engloba medidas específicas del lector (motivación, conocimientos y experiencias) y de la tarea (propósito).

2.3.2 Las operaciones de simplificación de textos

En un estudio llevado a cabo sobre un corpus paralelo formado por textos en español y sus equivalentes simplificados se identificó 8 operaciones de simplificación: sustitución, eliminación, inserción, separación, aproximación, reordenamiento, selección y unión (Saggion y Bott, 2011). Las autores describieron cada una de ellas.

Una de las operaciones más comunes, la **sustitución**, consiste en reemplazar palabras pocos frecuentes por otras de uso más común. Si bien normalmente se trata de una operación sencilla de implementar, enfrenta complicaciones si aparecen palabras compuestas en el texto.

La **eliminación** puede actuar en palabras, frases u oraciones. Frecuentemente se eliminan adjetivos y adverbios, ya que en la mayoría de casos no aportan información relevante al texto. No obstante, no siempre es posible llevarla a cabo porque en algunos casos puede ocasionar la pérdida a nivel semántico.

La operación de **inserción** tiene el objetivo de clarificar un texto, ya sea por medio de la definición de términos o la recuperación de partes eliminadas del mismo. En muchas ocasiones, se trata de una operación difícil de predecir y, por tanto, difícil de automatizar.

Por otra parte, la operación de **separación** tiene una gran importancia en la simplificación de textos en español. Esta consiste en separar oraciones largas y complejas (por lo general, oraciones compuestas) en otras más cortas. La dificultad

que se experimenta al implementar esta operación radica en la recuperación del sujeto de la oración luego que ésta ha sido separada.

El mecanismo de **aproximación** consiste en acercar psicológicamente el texto al lector. Por ejemplo: reemplazar la frase “en esta ciudad” por “en nuestra ciudad”. Los autores reconocen que esta operación es difícil de implementar debido a que requiere de conocimientos y mecanismos de inferencia del editor (Saggion *et. al.*, 2014),

El **reordenamiento**, como indica su nombre, consiste en cambiar el orden en el que la información es presentada al lector. Con frecuencia, se reduce a cambiar la posición de un verbo de reporte en la oración: en los textos éste suele ubicarse al final de la cita; luego de aplicar la operación, ésta se traslada al inicio de la misma.

Las operaciones de **selección** y **unión** son significativamente menos importantes. La operación de selección toma una frase nominal y la usa como encabezado; la de unión, junta dos piezas de información en una sola.

2.3.3 Analizadores sintácticos y léxicos

La estructura de un sistema de simplificación es muy variada, de acuerdo a las particularidades de cada lengua; no obstante, existen dos módulos que casi siempre están presentes: el sintáctico y el léxico (Saggion y Bott, 2014). La descripción de un sistema de simplificación de textos en español escapa del alcance de esta sección y será descrito en el Estado del Arte; no obstante, sí se explicará qué función desempeña cada uno de estos módulos.

En el marco de la simplificación de textos, el analizador sintáctico es el componente del sistema que se encarga de llevar a cabo la simplificación sintáctica del texto recibido. En él, la estructura sintáctica es representada por medio de un árbol de dependencia. Esto le facilita en gran medida llevar a cabo la operación de separación de oraciones anteriormente descrita. Sin embargo, cuando el sistema es basado en reglas, con frecuencia se obtienen árboles incorrectos y, por tanto, resultados incorrectos (Saggion *et. al.*, 2012).

Un árbol de dependencia es una estructura de flujo simple que consiste en un conjunto de nodos de decisión y hoja. El nodo de decisión inicial recibe el nombre de raíz y a partir de él se recorre la estructura hasta llegar a una hoja pasando por los nodos de decisión. La trayectoria a seguir para llegar a la hoja dependerá de la secuencia de decisiones que se tome en los nodos (Bird *et. al.*, 2009).

Por otra parte, el analizador léxico es el componente del sistema que realiza la simplificación léxica. Para ello, normalmente recurre a la operación de sustitución con la cual reemplaza palabras estadísticamente poco usadas por otras más comunes (Saggion *et. al.*, 2012). Otras estrategias usadas para la simplificación léxica incluyen la simplificación de números, sustitución de adjetivos étnicos y sustitución de verbos de reporte (Saggion *et. al.*, 2013).

2.4 Aprendizaje de Máquina

Para resolver un problema en una computadora se necesita un algoritmo: una secuencia de instrucciones que le indican cómo transformar los datos de entrada en datos de salida. Sin embargo, en algunas ocasiones no se cuenta con un algoritmo disponible para realizar algunas tareas; por ejemplo, la clasificación de correos electrónicos como basura o no. En tales casos se busca que la computadora "deduzca" por si sola el algoritmo para llevar a cabo la tarea en base a datos históricos. Lo anteriormente descrito constituye la esencia del aprendizaje de máquina (Alpaydin, 2010).

2.4.1 Agrupamientos, clasificación y tipos de aprendizaje

Dos problemas típicos abordados en el campo son la clasificación (o análisis de discriminantes) y el agrupamiento (o *clustering*, su término anglosajón):

- El **agrupamiento** es la organización de un conjunto de patrones (representados como puntos en un espacio multidimensional) en grupos basados en las semejanzas entre sus elementos.
- La **clasificación** es la tarea de asignar una etiqueta de clase correcta a un dato de entrada (Bird *et. al.*, 2009). Se trata de la metodología más importante en el campo de aprendizaje de máquina y es de vital importancia en el procesamiento de imágenes (Cunningham *et. al.*, 2008).

Resulta de importancia mencionar que el problema de clasificación y el problema de agrupamiento tienen una diferencia importante: el primero corresponde a un problema de aprendizaje supervisado y el segundo a uno de aprendizaje no supervisado. En el **aprendizaje supervisado** se dispone de una colección de patrones con sus respectivas etiquetas para entrenar el modelo; el problema consiste, entonces, en la asignación de la etiqueta correcta a un patrón sin etiqueta (JAIN *et.al.*, 1999). En el caso del **aprendizaje no supervisado**, sólo se dispone de

datos sin etiquetar; por tanto, el objetivo es encontrar regularidades entre ellos (Alpaydin, 2010).

2.4.2 Marco de trabajo para la clasificación supervisada

El primer paso en la creación de un clasificador es la decisión de cuáles son las características de los datos de entrada que se van a utilizar en la clasificación. Este paso incluye también la decisión de cómo representar dichas características (Bird *et. al.*, 2012). Esto resulta de vital importancia porque con frecuencia tiene un gran impacto sobre la probabilidad de extraer un buen modelo. Asimismo, es necesario mantener un balance en el número de características: si son muy pocas, el modelo obtenido no tendrá gran capacidad predictiva; si son muchas, el modelo reflejará particularidades del corpus de entrenamiento, un problema conocido como *overfitting* (Bird *et. al.*, 2012).

El paso siguiente es la codificación de un extractor de las características seleccionadas. Estas funciones devuelven diccionarios que asocian cada característica con su valor extraído del corpus (Bird *et. al.*, 2012).

Cuando el extractor de características ya ha sido codificado, se procede a aplicarlo sobre el corpus de entrenamiento. El conjunto de características extraídas se envía al algoritmo de aprendizaje de máquina, el cual genera el modelo del clasificador (Bird *et. al.*, 2012). Todo el proceso de entrenamiento se puede apreciar en la figura 1.1 (a).

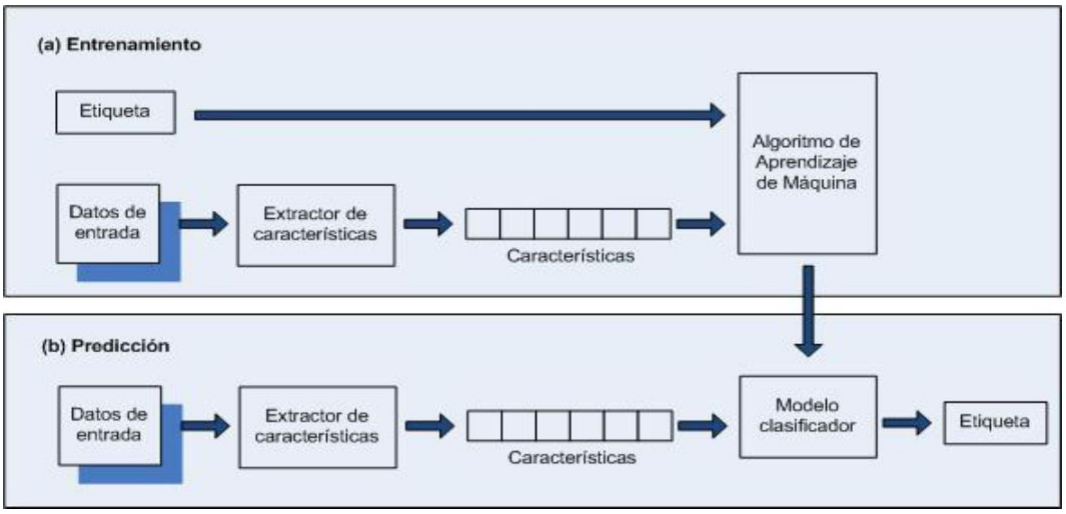

Fig. 1.1. Esquema del marco de trabajo para la clasificación supervisada.
Fuente: Vargas (2013).

Como se puede observar en la figura 1.1 (b), cuando el modelo ya está generado sólo necesita recibir el conjunto de características de una entrada para devolver como respuesta la etiqueta que le correspondería (Bird *et. al.*, 2012).

Una práctica recomendada para refinar el conjunto de características es llevar a cabo un análisis de errores. Para ello, es necesario dividir el corpus en dos: un conjunto de desarrollo y otro de prueba. El conjunto de desarrollo se divide nuevamente en dos: una colección para entrenamiento y una para las pruebas de desarrollo. Es con la colección para pruebas de desarrollo que se efectúa el análisis de errores. Asimismo, se puede inferir que para llevar a cabo lo anteriormente descrito, el corpus debe ser grande (Bird *et. al.*, 2012). Esto puede visualizarse en la figura 1.2.

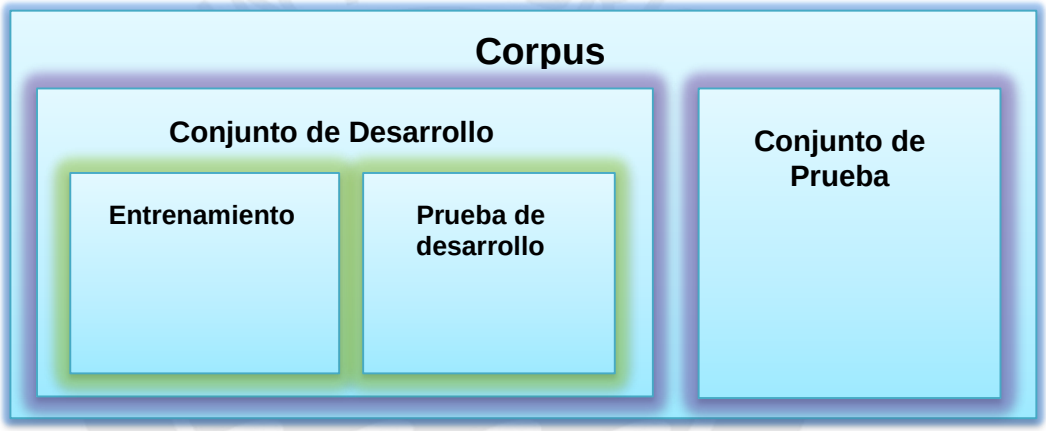

Fig. 1.2. División del corpus necesaria para llevar a cabo un análisis de errores.
Fuente: propia, adaptada de Bird *et. al.* 2012.

2.4.3 Evaluación del aprendizaje

Para saber si un modelo de clasificación es capaz de capturar correctamente un patrón se hace necesaria la existencia de métricas para evaluarlo. Estas métricas se obtienen del llamado conjunto de prueba, el cual contiene una parte del corpus como se vio en la sección anterior y en la figura 1.2. La idea es hacer pasar el contenido de esta colección por el clasificador para comprobar que éste le asigne la etiqueta correcta. Es muy importante que este conjunto sea diferente al de entrenamiento. De caso contrario, no se estaría probando nada (Bird *et. al.*, 2012).

La primera métrica a mencionar es la **exactitud**. Se trata de la más simple de todas las métricas usadas para evaluar un clasificador. Mide el porcentaje de datos

de entrada que el clasificador ha etiquetado correctamente. Es decir, su fórmula está dada por:

$$\boldsymbol{Exactitud} = \frac{\#exitos}{\#mediciones} \quad \ldots(\boldsymbol{Ec.1.2})$$

Por ejemplo, si un clasificador predice correctamente la etiqueta 60 veces en un conjunto de prueba de 80 elementos, su exactitud será del 60/80=75% (Bird *et. al.*, 2012).

Sin embargo, esta métrica es engañosa porque depende mucho del conjunto de textos que se usa para la evaluación. Por ejemplo, si se busca la palabra *banco* en una colección de prueba de textos financieros es muy probable que tenga una medida alta de exactitud; sin embargo, ésta no se puede atribuir al clasificador (Bird *et. al.*, 2012).

Para comprender las siguientes dos métricas, es necesario tener en cuenta que cuando se llevan a cabo tareas de clasificación (búsqueda de la etiqueta correcta o relevante) hay 4 posibles resultados:

- **Positivos verdaderos (TP):** ítems marcados como relevantes e identificados correctamente.
- **Falsos verdaderos (TN):** ítems marcados como irrelevantes e identificados de forma correcta.
- **Falsos positivos (error de tipo I, EI):** ítems irrelevantes que fueron identificados como relevantes.
- **Falsos negativos (error de tipo II, EII):** ítems relevantes que fueron identificados como irrelevantes.

La **precisión** es una métrica que indica cuántos de los ítems relevantes han sido identificados correctamente como tales. Su fórmula es:

$$\boldsymbol{Precision}\ (\%) = \frac{TP}{TP + EI} \times 100\% \quad \ldots(\boldsymbol{Ec.1.3})$$

La otra métrica es llamada **memoria** (*recall*) e indica el porcentaje de ítems relevantes que fueron identificados. Su fórmula es:

$$\boldsymbol{Memoria}(\%) = \frac{TP}{TP + EII} \times 100\% \quad \ldots(\boldsymbol{Ec.1.4})$$

Adicionalmente, (Bird et. al., 2012) describe una técnica usada frecuentemente en el campo llamada **validación cruzada** (*cross-validation*). Esta consiste en

subdividir el corpus original en N subconjuntos llamados pliegues. Enseguida, se entrena cada modelo usando todos los datos, excepto los del pliegue i-ésimo. Luego, se corre el modelo en dicho pliegue. Este procedimiento se repite N veces para cada pliegue. Si bien el resultado de cada uno de estos experimentos no es representativo por trabajar sobre muestras pequeñas, el resultado combinado sí es significativo y confiable.

3 Estado del arte

En esta sección, se describirá el estado actual de las herramientas e investigaciones relacionadas con el problema de simplificación textual en español. Se presentarán algunas soluciones diseñadas para la simplificación de textos en otros idiomas; en particular, en inglés y portugués.

3.1 Simplificación de textos en Español y el Proyecto Simplext

Como se mencionó en la sección del marco conceptual, la investigación en el campo de la simplificación de textos en español es limitada. Previo al proyecto Simplext, no se encontró una investigación de envergadura similar (Saggion y Bott, 2014). Es más previo al mencionado proyecto, no existía un corpus en español de textos simplificados (Saggion y Bott, 2011).

Simplext fue un proyecto promovido por el Ministerio de Industria, Turismo y Comercio del Gobierno de España cuyo objetivo fue desarrollar una solución de simplificación de textos en español para hacer accesible textos a personas con discapacidades cognitivas (Saggion et. al., 2011).

Con tal fin, se desarrolló un sistema de simplificación de textos en español con dos módulos: uno sintáctico y otro léxico. Debido a que hay una variedad de operaciones de simplificación, decidieron enfocarse en las operaciones de separación sintáctica y de sustitución léxica (Saggion y Bott, 2014).

El módulo léxico, que implementó la operación de sustitución léxica, fue el primero en ser diseñado por los autores y su desarrollo puede ser seguido en diferentes trabajos suyos, en particular, los trabajos de (Drndarevic y Saggion, 2012) y (Saggion et. al., 2013).

Una pregunta que puede surgir es: ¿bajo qué condiciones el módulo sustituye una palabra por otra? En un trabajo previo, (Drndarevic y Saggion, 2012), llegaron a la conclusión que un buen factor de decisión es la combinación de la longitud de palabra con su frecuencia. Para determinar la frecuencia de cada palabra

emplearon un diccionario basado en el Corpus de Español Actual. Por otra parte, con el fin de mejorar los resultados devueltos por el módulo, (Saggion et. al., 2012) propusieron la implementación de un filtro estadístico de resultados. Este filtro estadístico consistió simplemente en un clasificador binario que decidía si se debía efectuar o no una operación sobre una oración en base al tipo de sentencia relativa. De esta manera, sólo un tipo de sentencia relativa podía pasar el filtro (Saggion *et. al.*, 2012).

Según (Saggion et. al., 2013), las reglas de transformación implementadas para el módulo de simplificación léxica se agruparon en 4 grupos:

- **Expresiones numéricas:** se incluyeron operaciones como redondeo, pérdida de precisión, eliminación de detalles innecesarios de fechas, etc.
- **Expresiones en paréntesis:** se asumió que no aportan contenido esencial a la oración, por lo que son eliminados.
- **Adjetivos étnicos:** se reemplazaron los adjetivos étnicos por construcciones de la forma *[de + <Origen>]*.
- **Verbos de reporte:** se sustituyeron por la conjugación adecuada del verbo *decir*.

En el módulo de simplificación sintáctica se implementó la operación de separación de oraciones. Esta operación se implementó en base a reglas y trabajó con árboles de dependencia sintáctica (Saggion et. al., 2013). El mayor problema que se encontró fue manipular el árbol en sí y obtener un resultado gramaticalmente correcto (Saggion y Bott, 2014).

El funcionamiento de este sistema se basó en la aplicación secuencial de ambos módulos sobre un texto. Entre los criterios que se evaluaron se encontraron la gramaticalidad, preservación de significado y el grado de simplificación. Para ello, los textos producidos por el sistema fueron evaluados por humanos. Este grupo reconoció un gran porcentaje de errores en los textos producidos por el sistema. Los investigadores concluyeron que la mayor parte de dichos errores provino del analizador sintáctico y que tuvieron su origen en los árboles sintácticos usados para representar las oraciones. Dichas inconsistencias en los árboles fueron atribuidas al hecho que en muchos casos la gramática diseñada no fue perfecta y presentó ambigüedades que no fue capaz de resolver (Saggion et. al., 2013).

Para apalear este problema, los investigadores trabajaron en filtros estadísticos implementados como clasificadores binarios para la simplificación de oraciones con cláusulas relativas, obteniendo resultados diversos debido a que en el corpus que usaron dicha operación no era muy frecuente (Saggion et. al., 2013). Adicionalmente, investigaron por separado la eficacia de clasificadores binarios para tomar la decisión de separar o eliminar oraciones. No obstante, no se ha encontrado trabajos posteriores que indiquen que se hayan integrado al sistema desarrollado previamente (Saggion et. al., 2013-b).

3.2 Simplificación de textos en Inglés

La literatura está llena de intentos de simplificar textos en inglés por medio del uso de reglas sintácticas dirigidas a separar oraciones largas y complicadas en otras más cortas. Los trabajos de Chandrasekar (1996), Carrol (1999), Siddharthan (2004), Vickrey y Koller (2008) siguieron este lineamiento. Trabajos más recientes, como el de Yatskar (2010), exploraron métodos basados en datos para aprender las reglas de simplificación a partir del corpus (Woodsend y Lapata, 2011-b).

Un factor que facilitó el desarrollo de investigaciones de métodos basados en datos para la simplificación de textos en inglés fue la aparición la versión simplificada de Wikipedia (en inglés) en el 2001, compuesta por la simplificación de sus artículos. En consecuencia, se puso a disposición de la comunidad un gran corpus paralelo para las investigaciones en este idioma (Woodsend y Lapata, 2011).

Woodsend (2011) describió el desarrollo de un sistema de simplificación de textos en Inglés basado en un modelo de programación lineal entera. A diferencia de trabajos anteriores, el sistema no se basó en reglas de simplificación sino que las aprendió en base a oraciones que se extrajeron del corpus paralelo. Es más, el conjunto de entrenamiento fue tomado a partir del historial de revisiones de la Wikipedia Simple (Woodsend y Lapata, 2011-a; Woodsend y Lapata, 2011-b).

El sistema descrito por trabajó a nivel de oraciones: convirtió cada una que recibió como dato de entrada en una versión más simple. Para ello, la descompuso en sus componentes. Luego, simplificó (a nivel léxico y estructural) cada uno de ellos mediante las reglas de la gramática. Enseguida, generó todas las posibles simplificaciones que podrían resultar de la combinación de los componentes simplificados y escogió la mejor por medio del uso de la

programación lineal entera (Woodsend y Lapata, 2011-b). Las restricciones del modelo fueron: el uso de estructuras simples, el uso de palabras comunes y la gramaticalidad de la oración resultante (Woodsend y Lapata, 2011-a).

Como parte de evaluación de resultados, los textos simplificados se sometieron a juicio humano. Estos fueron evaluados en 3 niveles: simplicidad, gramaticalidad y conservación de significado. Cada uno de ellos fue calificado en una escala del 1 al 5, donde el 5 era el mayor calificativo. Los resultados de estas pruebas se resumen en la tabla 1.2. Lo anterior, en conjunto con una serie de pruebas de evaluación automática permitió concluir a los investigadores que su modelo fue el que tuvo un mejor desenvolvimiento: produjo textos más sencillos, coherentes gramaticalmente y con un significado igual o similar al de la oración original (Woodsend y Lapata, 2011-a).

Tabla 1.2. Resultados de la evaluación humana de textos.

Modelo	Simplicidad	Gramaticalidad	Significado
Wikipedia Simple	3.74	4.89	4.41
Zhu	2.92	3.42	3.44
Woodsend y Lapata	**3.64**	**4.55**	**4.19**

Fuente: Adaptado de Woodsend y Lapata. 2011-b.

Por otro lado, en la investigación de Zhu (2010), se trató el problema de simplificación textual como una forma especial de traducción en la cual la versión simplificada constituye la versión traducida del texto. Esta investigación utilizó un modelo de simplificación basado en árboles y otro que le permitió procesar el gran corpus paralelo de Wikipedia (Zhu *et. al.*, 2010).

De forma casi paralela, Yatskar (2010) exploró métodos la posibilidad de aprender reglas de simplificación léxica de textos a partir del historial de revisiones de Wikipedia. La principal complicación de este enfoque provino de la naturaleza misma de este historial de ediciones: no todas las ediciones que aparecieron en dicho historial provinieron de operaciones de simplificación, sino también de otras operaciones triviales (Yatskar *et. al.*, 2010)

3.3 Simplificación de textos en Portugués y PorSimples

A diferencia de lo que ocurre para el inglés, para el cual existen recursos para investigaciones (por ejemplo, una base de datos con las palabras más frecuentes del idioma, un corpus paralelo como el conformado por Wikipedia y su versión

simplificada, entre otros), para el Portugués tal facilidad de acceso a recursos de simplificación textual es inexistente (Candido y Copestake, 2011).

Con estas limitaciones, en noviembre del 2007 comenzó el proyecto PorSimples, una iniciativa para el desarrollo de tecnologías que simplifiquen el acceso a la información para individuos con un nivel de comprensión lectora baja y otras discapacidades que les impida comprender un texto. Para ello, se investigó una serie de campos diversos como la simplificación sintáctica, la simplificación léxica, la elaboración de textos y la agregación de textos (Aluísio y Gasperin, 2010).

Por la falta de recursos disponibles en este idioma, los participantes del proyecto crearon todas las herramientas necesarias para el proyecto. Así, elaboraron un manual para la simplificación de textos en portugués, crearon 9 *corpus* de textos de noticias y artículos científicos y recolectaron en un diccionario las palabras más usadas del idioma (Aluísio y Gasperin, 2010).

En lo que respecta a simplificación sintáctica, en el proyecto se desarrolló un módulo de análisis sintáctico basado en reglas junto a un algoritmo de aprendizaje de máquina que determinaba cuándo simplificar una oración (Specia, 2010). Entre las operaciones implementadas se encontraron: separación de oraciones, cambio de voz pasiva por voz activa y reordenamiento. Como era de esperarse, los textos resultantes tuvieron un mayor número de oraciones de menor longitud (Aluísio y Gasperin, 2010).

A partir del trabajo en PorSimples surgieron nuevas investigaciones en el campo. Un ejemplo es el trabajo posterior llevado a cabo por (Cándido y Copestake, 2011) cuyo objetivo fue la creación de una versión simple de Wikipedia en Portugués a la demanda (*on demand*). Para ello, se concentraron en las operaciones más difíciles de llevar a cabo para las personas con discapacidad de comprender textos: la separación de oraciones subordinadas y el cambio de la voz pasiva a voz activa. Para ello, se probaron las reglas planteadas sobre un corpus de 156 artículos. El resultado final fue satisfactorio: una precisión de 79.4% (Candido y Copestake, 2011).

En su tesis doctoral, Candido amplió su trabajo y formuló un modelo general para el análisis de lengua en sus diferentes niveles, al que llamó Molin; y un modelo de simplificación sintáctica, Sinsim. La idea del autor fue que el primero provea al segundo de la información necesaria para la simplificación a la vez que sirviera para validar los resultados de la simplificación. Esta integración, no obstante, no pudo

ser lograda por motivos que escapan del alcance de la presente revisión (Candido, 2013).

3.4 Conclusiones del Estado del Arte

A partir de lo presentado en esta sección es posible observar el estado actual de la investigación en simplificación textual en tres idiomas importantes: inglés, portugués y español. Un aspecto que vale la pena destacar es la abrumadora diferencia que existe en el estado de la investigación en simplificación textual en inglés comparado con los otros dos idiomas.

Un factor importante que ha permitido tal desarrollo ha sido la existencia de una versión simplificada de Wikipedia, la cual puso a disposición de los investigadores un *corpus* lo suficientemente grande como para utilizar métodos de aprendizaje de máquina en sus investigaciones (Woodsend y Lapata, 2011; Woodsend y Lapata, 2011-b). Esto último no puede ser llevado a cabo en otros idiomas debido a la inexistencia de un *corpus* significativamente grande. Así, por ejemplo (Candido y Copestake, 2011) reconocieron que su *corpus* de textos en portugués no era lo suficientemente extenso como para que su sistema pudiese aprender las operaciones de simplificación textual, por lo que se limitaron a usarlo para que su sistema aprenda a determinar cuándo debe ser simplificada una oración.

En Español, la investigación de (Saggion y Bott, 2014), en el marco del proyecto Simplext, fue una de las primeras en ser realizadas para este idioma. Ésta los condujo, entre otras cosas, a la elaboración de un módulo de simplificación sintáctica. Sin embargo, éste tenía un nivel significativo de errores debido a que fue diseñado en base a reglas, incluso para decidir qué operación de simplificación efectuar. Por ello, usaron filtros estadísticos para la separación de oraciones con cláusulas relativas. No obstante, no tuvieron resultados concluyentes debido a la poca frecuencia de dicha operación en el corpus. Adicionalmente, estudiaron por separado la capacidad de clasificadores binarios para las operaciones de eliminación y separación, pero no lo integraron al sistema.

De acuerdo a lo anterior, es posible encontrar una posibilidad de mejora para el módulo de simplificación sintáctica en español. En tal sentido, se espera adoptar un enfoque de aprendizaje de máquina para tomar la decisión de qué operación de simplificación aplicar (no solo eliminación y separación). Esto se lograría mediante clasificadores binarios integrados al módulo. En base a los resultados obtenidos para la simplificación sintáctica de textos en inglés, es posible anticipar que los

resultados serán iguales o mejores que aquellos obtenidos utilizando un enfoque basado en reglas. Asimismo, a partir de la investigación llevada a cabo en el proyecto PorSimples, es posible anticipar que no se requiere un corpus muy grande para entrenar al clasificador de decisión y obtener resultados positivos.

CAPITULO 2

1 Objetivo general

Implementar una herramienta de simplificación sintáctica que utilice un modelo supervisado de aprendizaje de máquina para la simplificación de textos en español.

2 Objetivos específicos

1. Formar un corpus paralelo de textos de género periodístico en español y sus versiones simplificadas.
2. Definir el conjunto de características que serán extraídas de los textos periodísticos a simplificar.
3. Formar un modelo de clasificadores que tome la decisión de cuáles operaciones de simplificación emplear.
4. Implementar el prototipo de una herramienta automática de simplificación de textos en español.

3 Resultados esperados

1. **Para el objetivo 1:** corpus formado por 190 noticias en español y las versiones de las mismas luego de aplicarles operaciones de simplificación.
2. **Para el objetivo 2:** funciones extractoras de características definidas, implementadas y probadas.
3. **Para el objetivo 3:** modelo de clasificadores binarios en serie que definan la operación de simplificación a ser ejecutada.
4. **Para el objetivo 4:** herramienta que aplica las operaciones de simplificación.

4 Herramientas, métodos y metodologías

El objetivo de esta sección es presentar, de una manera clara, las herramientas, métodos y procedimientos que se emplearán para la obtención de los resultados esperados. En vista que cada uno de ellos está vinculado al cumplimiento de un objetivo específico, es posible afirmar que dichos elementos (herramientas, métodos y procedimientos) llevarán al cumplimiento del objetivo del proyecto. De lo anterior se desprende la importancia que tiene esta sección. Con la finalidad de conseguir una mejor comprensión de cómo cada uno estos elementos será usado en el proyecto, en la tabla 2.1 se presenta un breve resumen de cuáles herramientas, métodos y/o procedimientos serán usados en la obtención de cada

resultado esperado. Posteriormente, se profundizará en cada método y herramienta mencionada en la tabla.

Tabla 2.1.: Mapeo de herramientas, métodos y procedimientos con resultados.

Resultado esperado	Herramientas, métodos y procedimientos a usarse
R.E.1: corpus formado por 190 noticias en español y las versiones de las mismas luego de aplicarles operaciones de simplificación.	Se conseguirá un corpus paralelo de noticias en español y equivalentes simplificados en coordinación con el equipo del proyecto Simplext. Se enriquecerá el corpus por medio de la **anotación de textos**, procedimiento que será facilitado con el **diseño e implementación** de una herramienta de ayuda.
R.E.2: funciones extractoras de características definidas, implementadas y probadas.	Se usará ***"PoS tagging"*** para apoyar la implementación de los extractores de características. Se usará el método ***"Leave one out"*** para seleccionar el subconjunto de atributos más adecuado para caracterizar el corpus.
R.E.3: modelo de clasificadores binarios en serie que definan la operación de simplificación a ser ejecutada.	Se utilizará la herramienta Weka para probar diferentes combinaciones de clasificadores utilizando el método ***de "10-fold cross validation"*** para analizar los resultados. Se usará experimentación numérica para comparar resultados.
R.E.4: herramienta que aplica las operaciones de simplificación.	Se implementarán las funciones que permitan simplificar textos. Se implementará una aplicación Web que simplifique los textos que reciba como entrada. **INFLESZ** es un programa que analiza métricas de textos. Entre ellas, el índice de Perspecuidad.

Fuente: propia.

4.1 Esquema general del proyecto

El propósito de esta sección es describir, a grandes rasgos, cuál es el esquema general del trabajo del proyecto. Por tal motivo, se postergarán las explicaciones de cada método o herramienta mencionada en esta sección. Además, se ofrecerá un mayor grado de detalle sobre la obtención de cada resultado esperado en los capítulos que correspondan a cada uno de ellos.

El primer paso será la obtención del corpus paralelo de noticias en español (R.E.1). Puesto que la creación de tal corpus escapa del alcance del presente proyecto, se coordinará con el grupo de investigación del proyecto Simplex para obtener su autorización de usar el corpus que utilizaron en sus investigaciones. Para que dicho corpus sea de utilidad para el proyecto, sin embargo, es necesario que tenga anotado las operaciones de simplificación que se llevaron a cabo. Por ello, se diseñará e implementará una herramienta que apoye dicha tarea.

Una vez que se disponga del corpus, será posible crear las funciones extractoras de características (R.E.2). Se utilizará "*PoS tagging*" para obtener información de cada palabra de una oración y así poder implementar la lógica de cada función de extracción (las cuales se seleccionarán tomando como punto de partida los trabajos de investigación en español y portugués) en base a la información que el método ponga a disposición.

Con las funciones extractoras de características implementadas y el corpus disponible se podrá utilizar la interfaz de Explorador de Weka para comparar diferentes clasificadores en la decisión de si llevar a cabo o no una operación (R.E.3). Se hará una experimentación numérica para comparar la efectividad proyectada de cada clasificador, la cual se obtendrá por medio de la validación cruzada ("10-*fold cross validation*"). Asimismo, en una segunda dimensión, se probará cada clasificador con un subconjunto de las características (método "*Leave one out*") con la finalidad de encontrar también el mejor subconjunto de características.

Conocido el subconjunto de características y el conjunto de clasificadores a usar será posible implementar la herramienta de simplificación sintáctica. En esta herramienta, se reconocen dos frentes: el *"back"* y el *"front"*. Para el *"back-end",* será necesario implementar las operaciones de simplificación que sean computacionalmente factibles. Para el *"front-end",* por otra parte, se implementará una aplicación Web que permita a un usuario introducir un texto y recibir como

respuesta el texto simplificado. Por tanto, el "*front*" y el "*back*" se comunicarán vía servicios web.

Para comprobar la efectividad del sistema, se utilizará un analizador de complejidad de textos llamado INFLESZ. Para tener una línea base sobre la cual comparar, se tomará la métrica de Flesch-Szigriszt (la cual fue explicada en la sección 2.3.1), a los textos originales y simplificados. En base a esta métrica, podrá clasificarse el texto en un rango que va desde muy complicado (valores bajos) hasta muy fácil (valores altos). Adicionalmente, podrá analizarse qué tanto se ha mejorado o empeorado la simplificación con respecto al proceso manual (corpus).

4.2 Métodos

En esta sección se describirá brevemente los métodos que se van a emplear en el desarrollo del proyecto.

4.2.1 Etiquetado ("PoS Tagging")

Se trata de un método que asigna una etiqueta a cada palabra que compone una oración (Mitkov, 2009). A pesar de ser menos informativo que un análisis sintáctico completo, provee de información de alto nivel de gran utilidad (como el tipo de palabra, su tiempo, conjugación, etc) que es usada para resolver problemas más complejos (Dale, Moisl, & Somers, 2000).

4.2.2 Dejar uno fuera ("Leave one out")

Método para seleccionar el subconjunto de características que debe usarse para alimentar un clasificador ya que con frecuencia el conjunto completo de características no produce resultados tan buenos como los primeros. El método consiste en probar el algoritmo de clasificación varias veces, dejando una característica fuera para ver el impacto que tiene. Puede combinarse con el entrenamiento del clasificador tomando una característica por vez (Gasperín *et.al.,* 2009).

4.2.3 Validación cruzada ("Cross validation")

Consiste en subdividir el corpus original en N subconjuntos llamados pliegues. Enseguida, se entrena cada modelo usando todos los datos, excepto los del pliegue i-esimo. Luego, se corre el modelo en dicho pliegue. Este procedimiento se repite N veces para cada pliegue. Si bien el resultado de cada uno de estos experimentos

no es representativo por trabajar sobre muestras pequeñas, el resultado combinado sí será significativo y confiable (Bird et. al., 2012).

4.3 Herramientas

En esta sección se describen brevemente las herramientas que se utilizarán para el desarrollo del proyecto.

4.3.1 WEKA

WEKA es definida por sus creadores, los miembros del Grupo de Aprendizaje de Máquina de la Universidad de Waikato, como un conjunto de algoritmos de aprendizaje de máquina que permite desempeñar tareas de minería de datos. Es una herramienta de libre distribución muy versátil; contiene herramientas para una gran diversidad de tareas como: pre-procesamiento de datos, clasificación, regresión, agrupación, asociación, visualización, etc. Al igual que FreeLing (que será tratado a continuación), es integrable a Java (WEKA, 2014).

4.3.2 FreeLing

FreeLing es un paquete que provee diferentes servicios de análisis de lenguaje en una variedad amplia de idiomas; entre ellos, el Español (FreeLing, 2012). Este paquete puede ser utilizado en Java y es de libre distribución. Algunas funcionalidades que ofrece son:

- Etiquetado de partes de discurso (*"PoS Tagging"*).
- Reconocimiento de entidades nombradas.

4.3.3 INFLESZ

INFLESZ es una herramienta computacional que produce índices lingüísticos que pueden ser usados para investigar la cohesión y coherencia de textos en español. Sus creadores la definen como un programa para evaluar la legibilidad de textos escritos (Legibilidad, 2007). Una de las métricas que utiliza para tal fin es el índice de Perspecuidad, que es precisamente la que se requiere para evaluar los resultados de la herramienta de simplificación (R.E.4).

4.4 Metodologías

4.4.1 Metodología para la Gestión del Proyecto

Un proyecto es un esfuerzo temporal llevado a cabo para crear un nuevo producto, servicio o resultado (PMBOK, 2013). De acuerdo con la definición propuesta por el *Project Management Institute* (en adelante, PMI), todo proyecto tiene un inicio y un fin en el cual el proyecto debe llevarse a cabo ya sea con éxito o no. Con la finalidad de aumentar las probabilidades de culminar un proyecto de forma exitosa, el PMI ha propuesto un marco de trabajo y mejores prácticas, el cual será adoptado y adaptado para el presente proyecto.

De acuerdo al PMBOK, en su última edición, hay 10 áreas de conocimiento que deben ser gestionadas en los proyectos. En el presente trabajo, se gestionará 5 de estas áreas de conocimiento. A continuación se listará y describirá cada una de ellas.

- **Gestión de Integración del proyecto:** incluye los procesos y actividades para unificar, definir, combinar y coordinar los múltiples procesos y actividades de gestión del proyecto. Es decir, se asegura que todas las áreas de conocimiento sean administradas de una manera coordinada (PMBOK, 2013). En el caso de presente proyecto, involucra la principalmente la elaboración del **acta de constitución del proyecto** (Anexo 1) y la **gestión del control de cambios** (Anexo 2).
- **Gestión de alcance:** su principal objetivo es asegurar que el proyecto incluya todo el trabajo requerido – no más – para completar con éxito el proyecto. Debe definir y controlar aquello que no estará incluido en el proyecto (PMBOK, 2013). Para este proyecto en particular, la gestión del alcance es importante para evitar llevar a cabo actividades que no son necesarias y que puedan llevar a destinar menos trabajo a actividades que sí son críticas para el proyecto. Por ello, se elaborará la **declaración del alcance del proyecto** (Anexo 3) y un **EDT y su diccionario** (Anexo 4).
- **Gestión del tiempo:** engloba al conjunto de procesos necesarios para definir el cronograma del proyecto y asegurar que éste sea culminado a tiempo. Es una consecuencia inevitable de la naturaleza temporal de un proyecto (PMBOK, 2013). El presente proyecto cuenta con un tiempo relativamente corto para ser culminado; por tanto, una correcta planificación

del mismo y el cumplimiento de los plazos de entrega serán críticos para asegurar su finalización en el tiempo estipulado. Para un control efectivo del tiempo, se incluye un **cronograma del proyecto** (Anexo 5).

- **Gestión de costos:** se controlará que el proyecto no exceda el presupuesto inicial. Se considerará como línea base los costos ficticios por recursos humanos (para mantener coherencia con las cifras usadas para controlar el tiempo). De esta forma, todo costo adicional podrá ser identificado y cuantificado. Se incluye el **costeo del proyecto** (Anexo 6).

- **Gestión de calidad:** en el caso de esta área de conocimiento, se controlará la calidad del proyecto en cuanto a su desenvolvimiento. Para ello, se controlará el avance del mismo usando el método de valor ganado; es decir, con las métricas CPI y SPI, cuyas especificaciones pueden revisarse en el **documento con la línea base de calidad del proyecto** (Anexo 7). Cabe mencionar que para poder aplicar este método, es necesario asignar un valor monetario a las horas-hombre trabajadas. En vista que en este proyecto no se lleva a cabo pago a personal, se ha fijado una tarifa ficticia de S/1.00 la hora del tesista y S/.1.50 la hora del asesor (para que sea un reflejo proporcional a las horas dedicadas).

- **Gestión de riesgos:** incluye los procesos que involucran el manejo, identificación, análisis y control de los riesgos de un proyecto. Su objetivo primordial es maximizar la posibilidad de ocurrencia de eventos positivos (oportunidades) y minimizar la probabilidad de ocurrencia de eventos negativos (riesgos). Está compuesto principalmente por siguientes procesos: planificación de la gestión de riesgos, identificación de riesgos, realizar análisis cualitativo, realizar análisis cuantitativo, planificar las acciones de respuesta y controlar los riesgos (PMBOK, 2013). Por ello, en la sección 10.3. se incluye la matriz de riesgos del proyecto.

4.4.2 Metodología para el Desarrollo

Por la cantidad de participantes en el proyecto (dos personas, contando al asesor) y la baja cantidad de requerimientos, se ha optado por emplear una metodología tradicional, en particular, una de cascada incremental iterativa. Se escogió esta metodología por su simplicidad. No se ha optado por un enfoque basado en *Rational Unified Process* (RUP) debido a que hay pocos requerimientos y sus probabilidades de sufrir cambios son bajas. De la misma forma, no se ha

optado por un enfoque ágil puesto que no se espera que los requisitos cambien frecuentemente y porque el número de participantes (2) es muy reducido como para justificar su elección.

Dentro del enfoque escogido, se llevará a cabo la etapa de especificación de requisitos del clasificador. Asimismo, se realizará el diseño e implementación de los extractores de características y la interfaz entre el clasificador y el usuario. Finalmente, mediante esta interfaz se llevarán a cabo las pruebas unitarias necesarias para asegurar el correcto funcionamiento de todos los elementos de la herramienta.

5 Alcance y Limitaciones

5.1 Alcance

El proyecto pertenece al área de Ciencias de la Computación; específicamente, a la rama de Procesamiento de Lenguaje Natural (PLN). Está concebido con una naturaleza exploratoria; en tal sentido, se analizará una solución alternativa para el problema de la simplificación de textos en español, en particular, a nivel sintáctico. Para ello, se hará uso de métodos de aprendizaje de máquina porque han dado resultados positivos cuando fueron empleados en proyectos de investigación de simplificación textual en inglés (Woodsend y Lapata, 2011) y portugués (Candido, 2013).

Por medio del proyecto se busca obtener un modelo que permita simplificar textos en español a nivel sintáctico con mejores resultados que los obtenidos por el grupo del proyecto Simplext, quienes elaboraron un módulo de simplificación sintáctica con un enfoque basado en reglas.

De la naturaleza probabilística de los algoritmos de aprendizaje de máquina, es posible deducir que el presente proyecto no busca dar una solución exacta al problema. En el caso particular del español, tal solución no es plausible de conseguir con la tecnología actual puesto que existen algunas operaciones de simplificación textual que son demasiado complejas como para ser automatizadas. Pertenecen a este último grupo aquellas que requieren de la experiencia del lector; por ejemplo: la distinción de lenguaje metafórico, la aproximación del texto a la realidad del lector, la inserción de adjetivos apropiados según el contexto, etc (Saggion y Bott, 2011).

Se trabajará con textos periodísticos en español recopilados durante la ejecución del proyecto Simplext y simplificados manualmente por editores especializados del

grupo DILES de la Universidad Autónoma de Madrid (Saggion *et. al*., 2014). Se ha escogido dicho género debido a su disponibilidad, facilidad de adquisición, variedad de temas y nivel moderado de complejidad. De esta manera, se podrá construir un corpus de entrenamiento que permita procesar textos no triviales y que no conduzca al problema de sobre-ajuste (*overfitting*).

Se trabajará con textos en Español por ser un idioma muy difundido en el mundo, cuyas reglas básicas ya son conocidas y dominadas por los participantes del proyecto. No obstante, uno de los principales motivos que llevó a la elección de este idioma es la escasez de herramientas que permitan llevar a cabo la simplificación textual en él. Es más, hasta lo que pudo ser encontrado en la revisión de literatura, solamente el grupo del proyecto Simplext (Saggion *et.al.*, 2011) ha trabajado en este campo, situación que resulta sorprendente si se tiene en cuenta el gran tamaño de población hispanohablante.

El corpus de textos recibido deberá ser anotado con las operaciones de simplificación que se llevaron a cabo. Para facilitar esta tarea manual, se diseñará e implementará una herramienta que apoye esta tarea como parte del proyecto. De la misma forma, se implementarán las funciones que permitirán extraer las características de los textos del corpus para posteriormente experimentar con los clasificadores.

Para que el producto sea de utilidad, es necesario que efectúe las operaciones de simplificación. Por ello, se incluye las respectivas implementaciones dentro del alcance del proyecto.

El trabajo en simplificación textual con frecuencia viene acompañado de la necesidad de llevar a cabo un análisis de complejidad. Es más, dicho concepto se encuentra incluido en la misma definición de simplificación textual; por ello, resulta natural la inclusión de elementos de análisis de complejidad en el proyecto. Si bien un análisis exhaustivo de complejidad escapa del alcance, sí es necesario realizar uno de alto nivel con la finalidad de evaluar la eficacia del clasificador. Por tal motivo, parte del proyecto involucra el uso de una herramienta que permita extraer la métrica de Flesch-Szigriszt para comparar el resultado de la simplificación automática contra la manual.

5.2 Limitaciones

Como todo esfuerzo humano, el presente trabajo tiene una serie de limitaciones que resulta conveniente hacer explícita con el fin de tenerlas presente durante el

desarrollo del mismo y para que futuros trabajos puedan encontrar mejoras sobre él.

La primera limitación identificable en el proyecto está impuesta por la misma naturaleza del clasificador: al utilizar un modelo de aprendizaje supervisado, su funcionamiento depende en gran medida del corpus que se utilice para su entrenamiento. La elección de las noticias que formarán parte de dicho corpus es subjetiva, por lo que el clasificador implementado reflejará en gran medida sus características y no las del universo de noticias en Español y mucho menos la de cualquier texto en Español. Esto es importante tener presente también porque implica que la herramienta no simplificará con el mismo nivel de eficacia textos que provengan de una fuente diferente, como libros u artículos científicos.

De manera similar a la limitación anterior, existe otra bastante relacionada: la selección de las características a extraer de los textos para proveer de información al clasificador es subjetiva. Por tanto, las características que se decida extraer serán las mejores que puedan ser encontradas al momento de realizar el trabajo, lo cual no implica que sean las mejores en el universo de características que pueden ser extraídas del texto. Esto quiere decir que pueden existir algunas que no sean imaginadas al momento de decidir las candidatas; o que algunas son mejores para corpus diferentes al recopilado en el presente trabajo.

5.3 Riesgos

Tal como se mencionó en el acápite 7.4.1, la gestión de riesgos es muy importante para incrementar las probabilidades de éxito de un proyecto. Parte importante para tal gestión es la identificación y evaluación de los mismos, que es lo que se realizará a continuación.

Igual importancia tiene idear medidas de mitigación que permitan disminuir las probabilidades de ocurrencia de los riegos negativos y medidas de contingencia para delinear el curso de acciones a seguir en caso el riesgo se haga realidad. El resumen de todo este proceso se encuentra en la tabla 2.2 que se muestra a continuación.

Tabla 2.2. Matriz de identificación de riesgos

Riesgo	Pr.	Impacto	Mitigación	Contingencia
Las noticias escogidas son de diferentes temas, pero de estructuras muy similares.	4	Clasificador sobre-ajustado para dicha estructura.	No aplica.	Agregar noticias y sus versiones simplificadas al corpus.
La curva de aprendizaje del uso de algunas de las herramientas es muy grande.	2	Retraso en la obtención de los R.E.2, 3 y 4.	-Lectura rápida de documentación. -Lectura de recomendaciones.	-Consultar a expertos detalles del uso de las herramientas. -Consultar en foros.
Conocimiento insuficiente de cómo efectuar las operaciones de simplificación sintáctica.	2	Reducción de operaciones que se pueden implementar.	Lectura de material bibliográfico de gramática española.	Consulta con especialistas.
La calificación obtenida de la herramienta de análisis de complejidad no tiene suficiente granularidad.	3	Se obtiene una amplia variedad de modelos posibles de clasificador, ya que la calificación proveniente de la herramienta no permite discriminar mayor detalle.	No aplica.	Medir métricas adicionales y combinarlas con el resultado del clasificador para tener mayor granularidad en el análisis de complejidad.
Pérdida de los datos e información de la tesis.	2	Retraso en todo el proyecto, pues tendría que rehacerse varios elementos.	-Uso de copias de resguardo. -Almacenamiento en la nube. -Almacenamiento en medios externos.	Recuperar la información por fragmentos.

Fuente: propia

6 Justificación y Viabilidad

6.1 Justificación

El acceso a la información es un derecho universal reconocido por la ONU. Esta información puede venir de diferentes formas: oral, escrita, digital, etc. La forma escrita, en particular, tiene una gran importancia ya que ha sido la forma en la que gran parte del conocimiento ha perdurado a lo largo del tiempo. Así, la lectura es una habilidad clave para el ser humano: es necesaria para el aprendizaje, para la adquisición de conocimientos de los sucesos de su entorno, para la realización de investigaciones, etc. Puesto que existe una gran cantidad de personas que tienen dificultades de lectura, la reducción de complejidad de textos representa un paso clave para cumplir el ideal que todas las personas tengan un acceso pleno a la información.

La simplificación de textos, no obstante, es una tarea compleja para el ser humano y, en algunos campos su aplicación no representa ningún beneficio. Por ejemplo, en el ámbito de empresas periodísticas, tal proceso resulta muy costoso (Drndarević y Saggion, 2012). Por tanto, la automatización de la simplificación de textos se puede convertir en un incentivo para que algunas de tales empresas la lleven a cabo. En este proyecto se busca proporcionar una herramienta que permita simplificar textos complejos a nivel sintáctico utilizando algoritmos de aprendizaje de máquina, un enfoque adoptado para tal fin en otros idiomas como inglés (Woodsend y Lapata, 2011a) y portugués (Candido y Copestake, 2011) con resultados positivos.

Este proyecto beneficia a toda persona con alguna disfuncionalidad lectora (como afasia y dislexia) o bajo nivel de comprensión lectora, toda vez que contribuye a disminuir la complejidad de un texto cuya comprensión pueda ser difícil debido a su estructura sintáctica. Ello les permitirá acceder a información para realizar sus investigaciones, tener conocimiento de lo que sucede en su entorno, los últimos avances en ciencias, etc. También beneficia a profesores y educadores, ya que les permite escalar la dificultad de los textos que utilizan en sus lecciones, de forma que éstos tengan la complejidad adecuada para el nivel de sus alumnos.

Adicionalmente, puesto que el proyecto se enfoca en la simplificación textual a nivel sintáctico, cabe la posibilidad de extenderlo en un trabajo futuro para que incluya la simplificación a nivel léxico. Es más, en tal caso, cabría la posibilidad de

integrar ambos productos en una sola herramienta de simplificación de textos en español.

6.2 Viabilidad

Luego de llevar a cabo el análisis de alcance, tiempo, costo y riesgos, es posible realizar el análisis de factibilidad del proyecto para determinar si es que es que se debería iniciar su ejecución o no. Por tal motivo, a continuación se llevará a cabo tal análisis.

El análisis del alcance del proyecto, que se llevó a cabo en la sección anterior, muestra que el presente trabajo tiene el nivel de complejidad adecuado para un proyecto de fin de carrera, ya que se propone un enfoque de solución no explorado en el campo de la simplificación textual en español, pero que sí ha sido aplicado con éxito en otros idiomas como inglés y portugués. Todo el trabajo del proyecto puede verse esquematizado en el Diagrama de Descomposición de Trabajo (EDT), que se encuentra en el Anexo 4 de este documento. Este diagrama muestra todo el trabajo que será llevado a cabo en el proyecto en forma de entregables, que para este diagrama son los resultados esperados de cada objetivo específico.

El análisis de tiempo del proyecto, cuya materialización se puede apreciar en el Diagrama de Gantt, adjunto en el Anexo 5 de este documento, demuestra que éste podrá ser llevado a cabo en un intervalo de 7 meses. En este tiempo, se irán completando los paquetes de trabajo del EDT según cronograma para llegar a los resultados esperados en el tiempo programado para asegurar la finalización del proyecto en el límite de tiempo fijado.

Durante la ejecución del proyecto, se utilizará herramientas de software libre para asegurar que el proyecto no involucre gastos para sus participantes. Esto es muy importante, ya que no se recibe apoyo financiero ni económico. No obstante, se cuenta con una reserva de contingencia y una reserva de gestión, como lo sugiere la metodología del PMI.

Por último, el análisis de riesgos, resumido en la tabla 2.2, demuestra que los riesgos que afronta el proyecto son manejables en cuanto es posible mitigar la mayoría y también afrontarlos en caso de su ocurrencia. Por otro lado, los riesgos que mayor impacto pueden tener sobre el proyecto, tienen una baja probabilidad de ocurrencia. Por tal motivo, es posible afirmar que en líneas generales, los riesgos son controlables.

Por lo anteriormente expuesto, se considera razonable afirmar que el presente proyecto es viable, por lo que las actividades que involucra deberían ser iniciadas lo antes posible.

CAPITULO 3

1 Corpus paralelo de noticias en Español

1.1 Origen y descripción del corpus

1.1.1 Origen del corpus

Para poder llevar a cabo el proyecto es necesario un corpus paralelo de textos en español. Este corpus debe estar formado por textos en español alineados con sus versiones simplificadas. La creación de un corpus con tales características requiere del trabajo de especialistas en el área de Lingüística, por lo que escapa del alcance de este proyecto. Por ello, se obtuvo la autorización para usar el corpus recopilado durante el desarrollo del proyecto Simplext (Saggion *et. al*., 2011; Saggion et. al., 2014), el único existente en Español con las características mencionadas.

Se trata de un corpus con 190 artículos de la agencia periodística Servimedia tomados de cuatro secciones: nacional, internacional, sociedad y cultura. La simplificación de estos artículos fue llevada a cabo por editores y revisada por expertos del grupo de investigación DILES de la Universidad Autónoma de Madrid. Ambos grupos siguieron las pautas indicadas en (Anula et. al., 2011).

Durante el proyecto Simplext, este corpus fue alineado utilizando un algoritmo no supervisado de alineación. Esta alineación fue mejorada posteriormente de forma manual con la ayuda de un editor de textos del framework GATE (Bott y Saggion, 2011).

1.1.2 Descripción del corpus

Como se mencionó anteriormente, el corpus está compuesto por 190 noticias. Si bien no es una gran cantidad de textos, cuando se hace la cuenta a nivel de oraciones se obtiene una muestra más grande: 1087 oraciones. Se trata de un dato importante ya que las operaciones de simplificación se llevan a cabo a nivel de oraciones, no de textos.

Algunas propiedades generales del corpus se resumen en la tabla 3.1. Puede observarse que cada tipo de noticia tiene una cantidad bastante similar de oraciones con la finalidad de evitar “favorecer” un tipo de noticia en particular. Asimismo, puede observarse que en promedio, se trata de noticias de pocas

oraciones (5.72 en promedio) pero complejas (30.06 palabras) y con una alta incidencia de oraciones compuestas.

Otro aspecto que no debe escapar de vista es que el número de oraciones por tipo de noticia es casi el mismo. Esta característica es muy importante para reducir un posible sobreajuste del modelo de clasificadores hacia un solo tipo de noticia.

Tabla 3.1. Propiedades generales del corpus

Propiedad	Valor
Número de oraciones	1087
Número promedio de oraciones por texto	5.72
Longitud promedio de oración (palabras)	30.06
Longitud promedio de oración (caracter)	170.39
Número promedio de signos de puntuación por oración	3.00
Número de oraciones en voz pasiva	36
Número de oraciones compuestas	466
Número de oraciones simples	621
Número de oraciones de “Nacionales”	282
Número de oraciones de “Internacionales”	279
Número de oraciones de “Sociedad”	247
Número de oraciones de “Cultura”	279

Fuente: propia

1.2 Procesamiento

Una vez recibido el corpus del grupo de investigación Simplext, se procedió a enriquecerlo con anotaciones manuales de las operaciones de simplificación que se llevaron a cabo sobre las noticias originales. Para ello, se diseñó e implementó una herramienta de anotación de textos. Esto se describe con mayor detalle en las secciones siguientes.

1.2.1 Objetivo

La necesidad de enriquecer el corpus con las operaciones de simplificación efectuadas nace del esquema de trabajo del aprendizaje supervisado explicado en el marco conceptual. Para poder usar el aprendizaje supervisado, cada entidad del conjunto de entrenamiento (el corpus) debe contar con su etiqueta correspondiente.

En el caso de los clasificadores binarios que se van a entrenar, la etiqueta corresponde a llevar a cabo la operación X (una operación por clasificador) o no. Por consecuencia, dicha información debe existir en el conjunto de entrenamiento. Sin embargo, el corpus compartido por el equipo de Simplext no la tiene, por lo que resulta necesario enriquecerlo con esos datos.

1.2.2 Operaciones de simplificación

Las operaciones de simplificación que se anotaron en el corpus son un subconjunto de aquellas reconocidas por el grupo de Simplext (Saggión et. al. 2011) y que se describieron en el marco conceptual. No obstante, luego de un análisis de las operaciones llevadas a cabo en PorSimples, se decidió incluir aquellas que podrían ser adaptadas al español.

A modo de resumen, las operaciones de simplificación anotadas fueron las siguientes:

- **Separación:** consiste en separar oraciones largas y complejas (por lo general, oraciones compuestas) en otras más cortas.
- **Reducción:** consiste en reducir el tamaño de las oraciones por diferentes mecanismos (con frecuencia, parafraseo). Se considera que ha ocurrido una reducción si entre oración original y simplificada existe una diferencia de 10 palabras.
- **Eliminación:** puede actuar en palabras, frases u oraciones. Frecuentemente se busca eliminar adjetivos y adverbios, ya que en la mayoría de casos no aporta información relevante al texto. También puede consistir en eliminar completamente una oración del texto.
- **Cambio:** dentro de esta categoría se agrupan varias subcategorías. Las más significativas son: procesamiento de expresiones numéricas, cambio de voz pasiva, cambio de sujeto de oración (por apositivos).
- **Reordenamiento:** ocurre cuando la oración se altera para que respete la forma Sujeto – Predicado.
- **Inserción:** tienen el objetivo de clarificar un texto, ya sea por medio de la definición de términos o la recuperación de partes eliminadas de la misma.
- **Mantener:** simplemente no se hace nada, deja la oración tal como era originalmente o con cambios menores (por ejemplo, parafraseo que no llegue a constituir una operación de reducción).

Para anotar estas operaciones, se elaboró una guía de anotación del corpus, la cual se puede consultar en el **Anexo 9**. La tabla 3.2 muestra un resumen estas operaciones junto con la frecuencia de cada una en el corpus.

Tabla 3.2. Operaciones anotadas en el corpus

#	Nombre operación	Origen	Frecuencia absoluta	Frecuencia relativa (%)
1	Eliminación	Simplext / PorSimples	470	31.25
2	Separación	Simplext / PorSimples	317	21.08
3	Reducción	Simplext	303	20.15
4	Mantener	PorSimples	245	16.29
5	Cambio	Simplext / PorSimples	127	8.44
6	Reordenamiento	PorSimples	30	1.99
7	Inserción	Simplext	12	0.80

Fuente: propia

Tal como puede observarse en la tabla anterior, la operación más frecuente es la eliminación (31%), seguida por la separación de oraciones (21%) y la reducción (20%). Asimismo, contra todo pronóstico, la operación de no hacer nada es bastante frecuente (16%). En base a ello, es posible inferir que los clasificadores que más impacto tendrán sobre el desempeño del sistema serán ellos, ya que decidirán si efectuar o no las operaciones efectuadas con mayor frecuencia por anotadores humanos.

1.2.3 Estructuras de datos

El corpus original estaba formado por un conjunto de archivos XML en el que cada noticia estaba representada en 2 archivos XML:

- 1 archivo con el texto original
- 1 archivo con el texto simplificado

Con la finalidad de poder identificar cada pareja de archivos, se siguió el siguiente patrón para el nombre de cada uno:

- Para la versión original: ***<id_texto><tipo_noticia>.full.xml.*** Por ejemplo: 01CULT.full.xml.
- Para la versión simplificada***: <id_texto><tipo_noticia>.simp.xml.*** Por ejemplo: 01CULT.simp.xml.

Se manejaron 4 tipos de noticia: nacional (NAC), internacional (INT), cultura (CULT) y sociedad (SOC). Por otro lado, la estructura interna del archivo fue la misma en ambos casos. Por ejemplo, para una de las noticias se tuvo:

<Simplext>

<Sentence *sno="0" alignedSents="0"*>

Alex de la Iglesia, Premio Nacional de Cine

</Sentence>

…

<Sentence *sno="5" alignedSents="6"*>

En casos excepcionales también podrá otorgar reconocimiento.

</Sentence>

</Simplext>

En esta estructura se reconocen las etiquetas que se encuentran en la tabla 3.3.

Tabla 3.3. Etiquetas del archivo XML.

Etiqueta	Atributo	Descripción
Simplext	Ninguno	Marca el inicio y fin de un texto.
Sentence	Sno	Indica el número de oración dentro del texto.
Sentence	AlignedSents	Indica el número de oración que corresponde a su versión simplificada dentro del archivo correspondiente. Si a una oración le corresponde más de una alineación, se tendrá una lista separada por el carácter ";".

Fuente: propia

Una vez anotado el texto con la herramienta de anotación desarrollada (y que se describirá más adelante), se generó nuevamente un archivo para que pueda servir para proyectos futuros. No obstante, se decidió que este archivo nuevo fuera un JSON debido a que es un formato más simple, no requiere llenar una etiqueta con atributos de cada operación anotada y permite serializar con mejor legibilidad objetos complejos como vectores. Así, por ejemplo, la versión anotada de un texto sería:

```
{ "id":1, "tText":"CULT", "oraciones":
        [       { "id" : { "idTexto":"1", "idOracion":"0", "tText":"CULT" } ,
                  "oración" : "Alex de la Iglesia, Premio Nacional de Cine",
                  "separacion" : "0" , "reducción" : "0" , "eliminación":"0" ,
                  "mantener" : "1", "cambio" : "0" , "reord" : "0", "insc" : "0"
                } ,
        … ]
}
```

Lo resaltante en estos archivos son los atributos correspondientes a las operaciones (separación, reducción, eliminación, cambio, reordenamiento, inserción y mantener): son atributos binarios en los que 1 significa que ocurrió la operación y 0 significa que no ocurrió.

Por otra parte, los archivos de extensión ARFF constituyen la fuente de datos para la herramienta Weka, con la que se va a probar una diversidad de clasificadores. Estos archivos contienen los datos y los metadatos del vector de características que se extrae de cada texto. Asimismo, contiene la etiqueta que se le asigna a cada vector. Un ejemplo de la estructura de dicho tipo de archivos puede encontrarse en la documentación de Weka (Weka, 2014) y es la siguiente:

@RELATION <nombre de relación>
@ATTRIBUTE <nombre del atributo> <nombre del tipo de dato>
@DATA
X11 X12 X13 X14 … X1n Etiqueta 1
X21 X22 X23 X24 … X2n Etiqueta 2
…
Xm1 Xm2 Xm3 Xm4…Xmn Etiqueta m

Donde Xmn es el valor de la característica n-ésima en el m-ésimo vector de características. Por otra parte, los tipos de datos que soporta son: *numeric* (*real* e *integer*), *string*, *date* y *nominal*.

1.3 Herramienta de apoyo para la anotación

Con la finalidad de agilizar la anotación manual del corpus, se diseñó e implementó una herramienta que facilita dicha anotación colocando el texto original vs el texto simplificado, dejando a criterio del usuario la selección de las operaciones efectuadas. Dicha herramienta lee los archivos XML del corpus original y genera los archivos JSON que se describieron en la sección anterior.

1.3.1 Análisis

Al tratarse de una herramienta de un par de pantallas, los requerimientos son pocos y se encuentran registrados en la tabla 3.4.

Tabla 3.4. Catálogo de requerimientos.

#	Requerimiento	Prioridad	Tipo
1	La herramienta debe poder leer archivos XML con la estructura definida en (12.2.2).	Alta	F
2	La herramienta debe permitir al usuario seleccionar las operaciones de simplificación efectuadas: separación, reducción, eliminación, cambio, reordenamiento, inserción y mantener.	Alta	F
3	La herramienta debe mostrar cada oración del texto original con su equivalente del texto simplificado de manera simultánea. Para ello, se asume que los nombres de archivo cumplen con la nomenclatura descrita en (12.2.2).	Alta	F
4	La herramienta debe poder exportar el resultado a un archivo JSON con la estructura definida en (12.2.2).	Alta	F
5	La herramienta debe permitir al usuario moverse entre las distintas oraciones de un mismo texto.	Media	F
6	La herramienta debe guardar automáticamente las operaciones seleccionadas por el usuario para las oraciones de un texto hasta que decide analizar uno nuevo.	Media	F
7	La herramienta deberá permitir al usuario configurar el directorio de origen y destino de los datos.	Media	F
8	La herramienta permitirá generar un archivo ARFF en base a los archivos JSON.	Alta	F
9	La herramienta será implementada en Java SE 7.0.	-	NF
10	La herramienta será implementada usando el patrón MVC.	-	NF

Fuente: propia

1.3.2 Diseño

Para especificar el diseño de la herramienta, se utilizará el modelo 4+1.

Vista de Casos de Uso

La figura 3.1 muestra el diagrama de casos de uso.

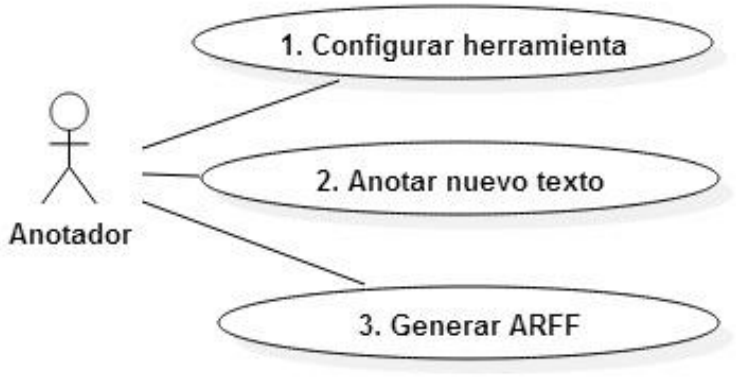

Fig. 3.1. Diagrama de casos de uso de la herramienta de apoyo para la anotación.
Fuente: propia.

Las especificaciones de estos casos de uso se muestran en las tablas 3.5 a 3.7.

Tabla 3.5. Especificación del caso de uso 1.

Código	1
Caso de uso	Configurar herramienta.
Actor	Anotador.
Descripción	Seleccionar directorio donde se guardaran los archivos JSON generados por la herramienta.
Precondición	No hay directorio donde guardar.
Flujo básico	1. En la pantalla, hacer clic en el botón "***Configurar***". 2. El sistema mostrará una ventana con dos selectores de directorios: uno para los archivos JSON y otro para los archivos ARFF. 3. Dar clic en uno de los botones y seleccionar la ruta de los archivos de la extensión indicada.. 4. Dar clic en el botón restante y seleccionar la ruta de los archivos de la extensión indicada. 5. Navegar hasta el directorio deseado. 6. Dar clic en el botón "***Abrir***". 7. La herramienta volverá a la pantalla principal.
Flujo alternativo	No tiene.
Postcondición	Directorio donde guardar definido.

Fuente: propia

Tabla 3.6. Especificación del caso de uso 2.

Código	2
Caso de uso	Anotar nuevo texto.
Actor	Anotador.
Descripción	Comparar una oración de un texto con su versión simplificada y, acto seguido, seleccionar las operaciones que se llevaron a cabo. Finaliza con la exportación de la anotación en un archivo JSON.
Precondición	No existe archivo JSON de salida en el directorio escogido en el caso de uso 1.
Flujo básico	1. En la pantalla, hacer clic en el botón "***Nuevo***". 2. El sistema mostrará una nueva ventana con el anotador. 3. Dar clic en el botón "***Buscar***". 4. Se abrirá un cuadro de texto para seleccionar el archivo XML que corresponde a la versión "original". Se asume que el nombre de archivo cumple con las condiciones expuestas en 12.2.2. 5. Se mostrará cada oración (una por una) del texto original y su equivalente en la versión simplificada. 6. Seleccionar las operaciones de simplificación que se hayan llevado a cabo para convertir el texto original en el simplificado. 7. Hacer clic en el botón de "<<", para regresar a la oración anterior o ">>", para avanzar a la oración siguiente. Repetir 5 a 7. 9. Al acabarse las oraciones del texto original, ya no se podrá seguir avanzando. En tal momento, hacer clic en "***Exportar***" para generar el archivo JSON en el directorio configurado en el caso de uso 1.
Flujo alternativo	Si se intenta abrir un archivo cuyo nombre no respeta el patrón indicado en 12.2.2, la herramienta mostrará un mensaje de error al usuario.
Postcondición	Aparece archivo JSON de salida en el directorio escogido en el caso de uso 1.

Fuente: propia

Tabla 3.7. Especificación del caso de uso 3.

Código	3
Caso de uso	Generar ARFF
Actor	Anotador.
Descripción	Toma un grupo de archivos JSON generados por el caso de uso 2 y genera un archivo ARFF a partir de ellos..
Precondición	No existe archivo ARFF de salida en el directorio escogido en el caso de uso 1.
Flujo básico	1. En la pantalla, hacer clic en el botón "***Generar ARFF".*** 2. Seleccionar los archivos JSON a partir de los cuales se va a generar el archivo ARFF. 3. Esperar que termine el proceso de exportación.
Flujo alternativo	Si ocurre algún error durante la exportación, se mostrará un cuadro de diálogo informativo al usuario.
Postcondición	Aparece archivo ARFF de salida en el directorio escogido en el caso de uso 1.

Fuente: propia

Vista de desarrollo

La figura 3.2 muestra el diagrama de componentes. Lo que hay que resaltar del diagrama es que el sistema recibe como entrada archivos XML (corpus) y devuelve archivos JSON o ARFF luego del procesamiento. También se puede observar en la figura que la herramienta sigue un patrón MVC.

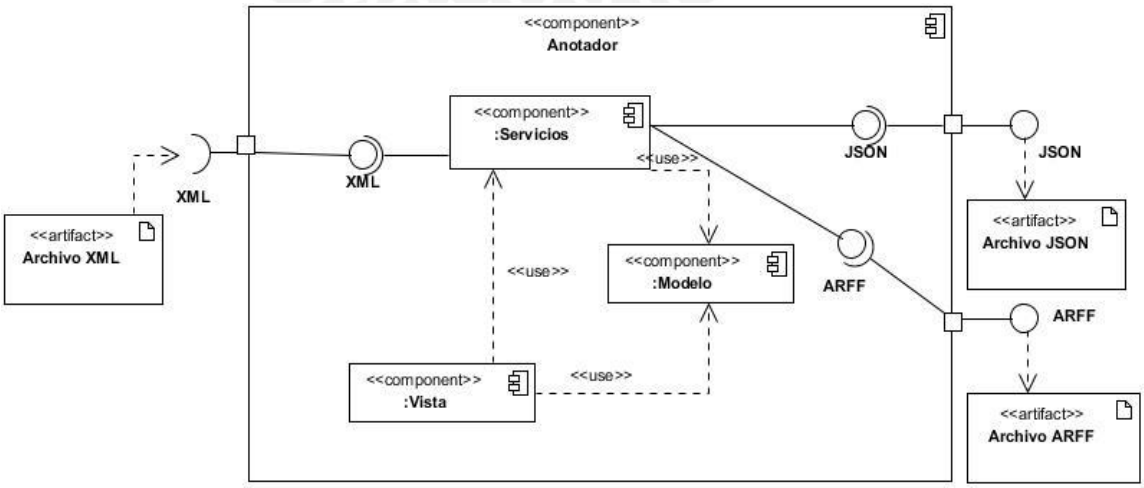

Figura 3.2. Diagrama de componentes de la herramienta.

Fuente: propia.

Vista Lógica

La figura 3.3, que se muestra a continuación, corresponde al diagrama de clases de la herramienta.

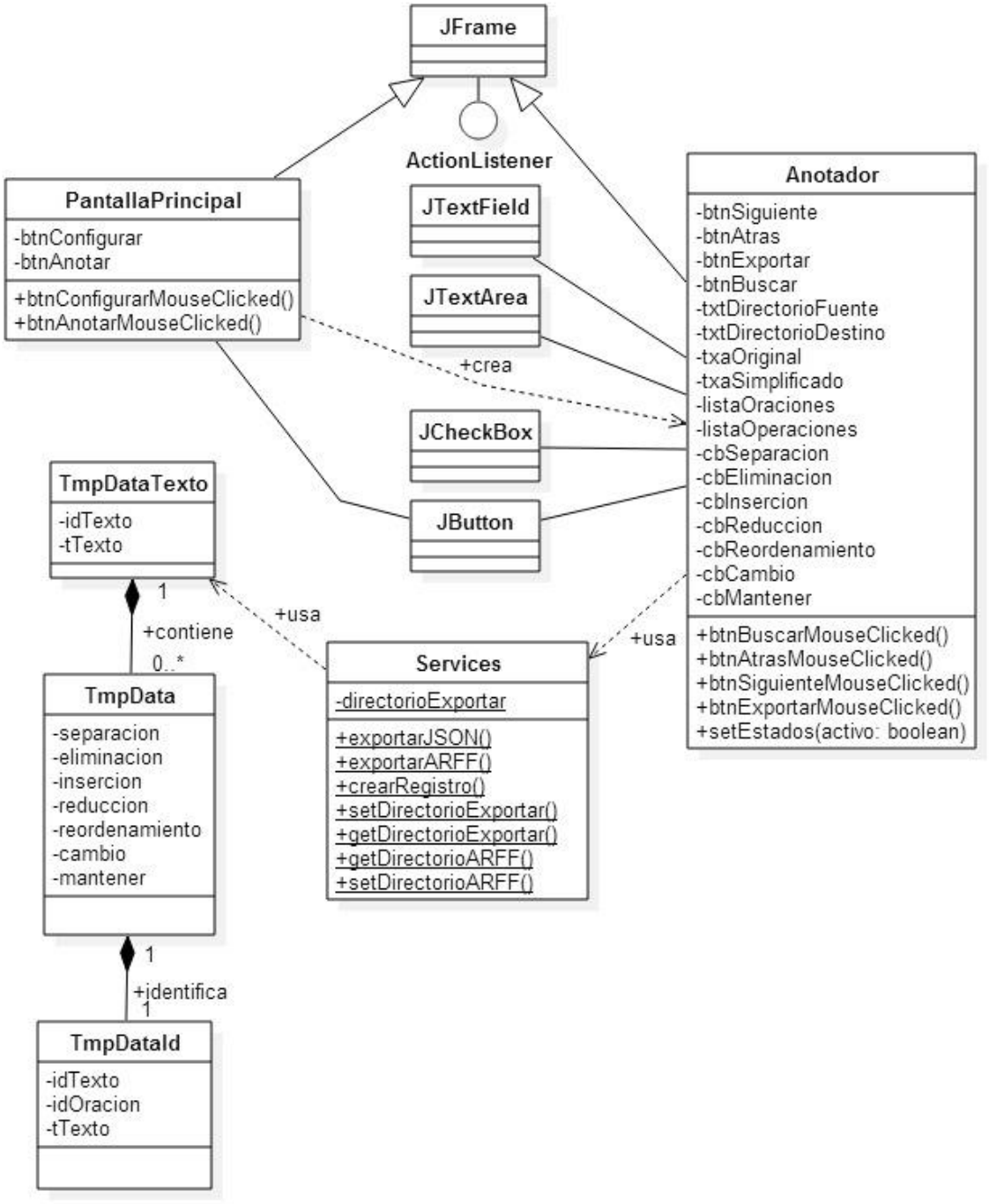

Figura 3.3. Diagrama de clases.
Fuente: propia.

Un detalle interesante de observar es la organización de las clases que corresponden al modelo (la herramienta ha sido implementada siguiendo un patrón MVC): un texto (noticia) está identificado y compuesto por varias oraciones, cada

una de las cuales almacena su propia información sobre qué operaciones de simplificación ha sufrido.

Vista de procesos

En la figura 3.4 se muestra el diagrama de actividades de la herramienta. Puede observarse que el flujo es bien sencillo: el usuario configura sus directorios y luego puede escoger entre anotar el corpus o generar el archivo de datos ARFF.

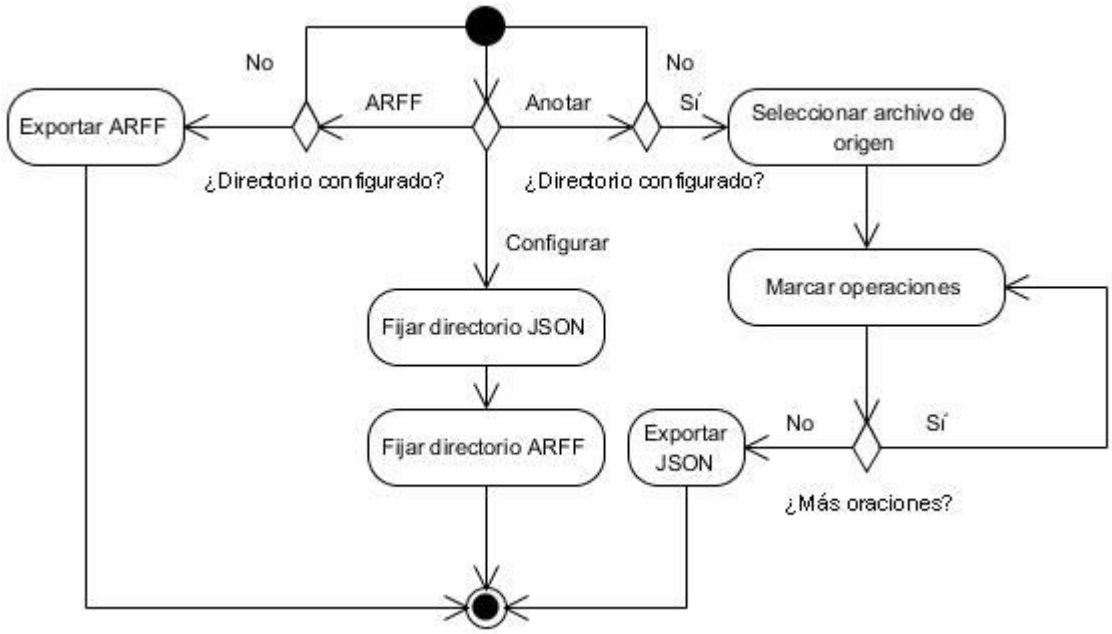

Figura 3.4. Diagrama de actividades de la herramienta de anotación.
Fuente: propia.

Cabe destacar que es necesario que el usuario haya configurado los directorios para que la herramienta le permita trabajar. Caso contrario, no podrá llevar a cabo ninguna de las otras dos operaciones (anotar y exportar).

Vista física

Esta vista es omitida debido a que la herramienta será desplegada sobre una PC y no necesita de otros elementos (como servidores o internet) para su funcionamiento.

1.3.3 Prototipo

Se implementó la herramienta usando Java como lenguaje de programación. Las figuras 3.5 y 3.6 muestran la apariencia de la herramienta y su funcionamiento.

Fig. 3.5. Pantalla inicial de la herramienta de anotación.
Fuente: propia.

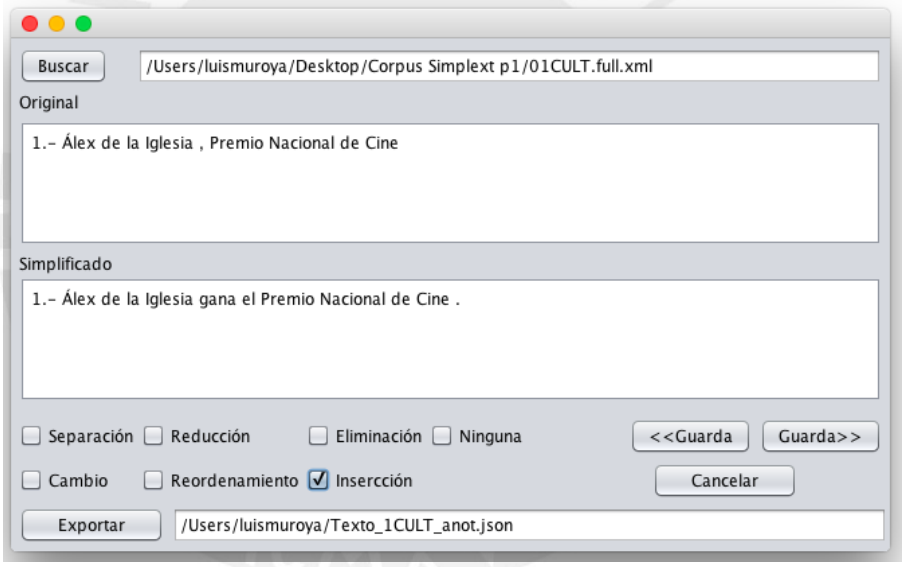

Fig. 3.6. Pantalla del anotador.
Fuente: propia.

En la figura 3.5 se puede reconocer la pantalla inicial que se abre cuando se inicializa la herramienta. En ella, se tiene la opción de configurar (caso de uso 1) o de hacer una nueva anotación (caso de uso 2).

La figura 3.6 muestra, por otra parte, la ventana de diálogo del anotador. Como se había mencionado previamente, hay un botón buscar que permite seleccionar el archivo XML del texto original. El nombre del texto simplificado es sencillo de obtener: es el mismo que el del texto original pero reemplazando la palabra “full” por

"simp", por lo que no es necesario que el usuario escoja este último de forma manual.

Asimismo, en la figura 3.6 se puede reconocer las dos grandes áreas de texto a través de las cuales se muestran de manera paralela el texto original y el texto simplificado. En la parte inferior se encuentra, por otro lado, las operaciones de simplificación que han sido consideradas y los botones de navegación ("***<<Guarda***", "***Guarda>>***").

Finalmente, en la parte inferior de la figura 3.6, se puede observar el botón "***Exportar***". Como dice su nombre, permitirá exportar todas las oraciones de un texto a un archivo JSON.

1.3.4 Detalles de implementación

Para la lectura de los archivos XML se tuvo que emplear tecnología JAXB para poder interpretarlos. Por tal motivo, se creó una plantilla XML (o esquema) que se adecuara a la estructura de los archivos del corpus. El archivo XSD resultante tiene la siguiente estructura:

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Simplext"> <xs:complexType> <xs:sequence>
        <xs:element name="Sentence" maxOccurs="unbounded" minOccurs="0">
          <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string">
                <xs:attribute type="xs:byte" name="sno" use="optional"/>
                <xs:attribute type="xs:string" name="alignedSents" use="optional"/>
                </xs:extension> </xs:simpleContent>
          </xs:complexType></xs:element></xs:sequence></xs:complexType>
  </xs:element>
</xs:schema>
```

1.4 Conclusiones del capítulo

A lo largo del capítulo se ha analizado la obtención y procesamiento del corpus. Al finalizar el mismo puede concluirse que el corpus recibido de parte del grupo Simplext cumple los requerimientos para el proyecto: contiene noticias en español, es lo suficientemente grande (en cantidad de oraciones) para llevar a cabo métodos

de aprendizaje supervisado, la distribución de los tipos de noticias es uniforme y cada oración del mismo está alineada con su versión simplificada.

Se pudo implementar con éxito una herramienta que permite anotar el corpus y devolver su resultado en un archivo de formato JSON o XML para su posterior uso y/o distribución. Gracias a la misma, se pudo anotar las noticias del corpus en poco tiempo.

Finalmente, en base a las frecuencias observadas de las operaciones, es posible concluir que las más importantes son la separación, la eliminación, la reducción y el cambio. Las operaciones de inserción y de reordenamiento son muy poco frecuentes, por lo que quedarán excluidas del módulo. Existe otro motivo para no considerar estas operaciones: generan un conjunto de entrenamiento con clases desbalanceadas, lo cual es un problema para el esquema de aprendizaje supervisado. Este problema se expondrá con mayor detalle en el siguiente capítulo.

CAPÍTULO 4

1 Funciones extractoras de características

En esta sección se describirá el proceso de definición, implementación y pruebas de las funciones necesarias para la extracción de las características de los textos que se procesan.

1.1 Selección de las características a extraer

Para seleccionar las características que se van a extraer de los textos se tomó como punto de partida un estudio llevado a cabo por (Stajner et. al., 2013) en el cual se evalúa el desempeño de clasificadores para predecir la operación de eliminación. Si bien constituye un primer paso para la adopción de un enfoque más estadístico que estático, no se observa indicios de la integración de este clasificador ni con el resto del sistema de simplificación ni con otros clasificadores. No obstante, en el estudio se utilizó un conjunto de características que fueron tomados como referencia para las características.

Por otra parte, por la similitud del portugués con el español, se decidió incorporar características que se extrajeron durante el proyecto PorSimples (Gasperín et. al., 2009). Lógicamente, se encontraron algunas que eran específicas del portugués y que no se pudieron adaptar al español.

Finalmente, algunas características fueron añadidas al conjunto en base a su necesidad para implementar alguna operación en particular o para calcular alguna métrica de interés sobre los textos del corpus.

La lista resultante se incluye en la tabla 4.1 que se muestra a continuación.

Tabla 4.1. Conjunto de características a extraer y sus abreviaturas y fuentes.

Grupo	N	Nombre	Abrev.	Simplext	PorSimp.
PoS Tagging	1	Adjetivos	adj	X	X
	2	Adverbios	adv	X	X
	3	Conjunciones Coordinadas	cco	X	X
	4	Conjunciones Subordinadas	csu	X	X
	5	Determinante	det	X	
	6	Gerundios	ger	X	
	7	Imperativos	imp	X	
	8	Indicativos	ind	X	
	9	Infinitivos	inf	X	
	10	Participios	par	X	
	11	Preposiciones	pre	X	
	12	Pronombres	pro	X	X
	13	Subjuntivos	sub	X	
	14	Sustantivos	sus	X	X
	15	Sustantivos Propios	spr		X
	16	Verbos	v	X	X
Sintácticos	17	Tam. de Frases Nominales	fraNom		X
	18	Tam. de Frases Preposicionales	fraPre		X
	19	Tam. prom. de Frases Verbales	freVeb		X
	20	Cláusulas coordinadas	cCoord		X
	21	Clausulas relativas	cRelat		X
	22	Cláusulas subordinadas	cSubor		X
	23	Frases adverbiales	fraAdv	X	
	24	Frases nominales	fraNom	X	X
	25	Frases preposicionales	fraPre		X
	26	Frases verbales	fraVeb	X	X
	27	Profundidad de arbol	profund	Propia	
	28	¿es oración compuesta?	compl	Propia	
	29	¿es voz pasiva?	vpasiva		X
Complejidad	30	Tamaño prom. de palabras en oración	promPalOra		X
	31	Numero prom. caracteres en oración	promCarOra	X	X
	32	Expresiones numéricas	exprNumOra	X	
	33	Palabras en oración	palabraOra	X	X
	34	Signos de puntuación	puntuacOra	X	
	35	Posición	pos	Propia	
	36	Sujeto primero	sujPrim	Propia	
	37	Porcentaje de repetidas	porcRep		

Fuente: propia.

De la tabla también puede rescatarse la tipificación de las características. Con la finalidad de manejar mejor el conjunto, este ha sido partido en tres grupos: las características básicas (PoS), las sintácticas y las de complejidad.

1.2 Implementación de los extractores de características

Para la implementación de las 37 funciones de extracción de características se utilizó el lenguaje Java y el apoyo de Freeling 3.1, que provee de un gran número de funcionalidades para el análisis lingüístico. La funcionalidad que se va a utilizar en este proyecto es la de etiquetado de partes de discurso (“PoS Tagging”). El paquete ha sido publicado bajo licencia GNU GPL, por lo que fue posible descargar el ejecutable y código fuente de la página oficial.

Cabe mencionar que Freeling está programado en C++, por lo que no puede usarse directamente en Java. No obstante, provee una API de Java que permite hacer uso de sus funcionalidades en dicho lenguaje de programación. Sin embargo, para compilar dicha API debe crearse un Makefile adecuado, tarea complicada en un entorno de Mac OS X ya que su sistema de archivos está organizado de forma diferente a como está organizado en Linux o Windows. Asimismo, fue necesario modificar algunas clases de C++ para que la API funcione correctamente en el entorno de Java.

Luego de un proceso de diseño, se optó por seguir la arquitectura que se describe en el diagrama de componentes de la figura 4.1 (siguiente página).

De la figura puede apreciarse que el analizador (conjunto de extractores) recibe del exterior la oración que se desea analizar a través del puerto “Oración”. Esto es muy interesante porque es a través de este puerto que puede acoplarse con el componente de Servicios de la herramienta de apoyo de anotación. De la misma forma, existe un puerto “OracionSalida” a través del cual el analizador devuelve al exterior el vector de características dentro del mismo objeto Oración.

También puede observarse que el acceso a Freeling por parte de los extractores no es directo, sino que se da a través del componente Módulo, el cual accede a Freeling por medio de la interfaz modFreeling.

Finalmente, es posible observar que se llevarán a cabo tareas en varios hilos gracias al uso de la clase “Thread” y la interfaz “Runnable”.

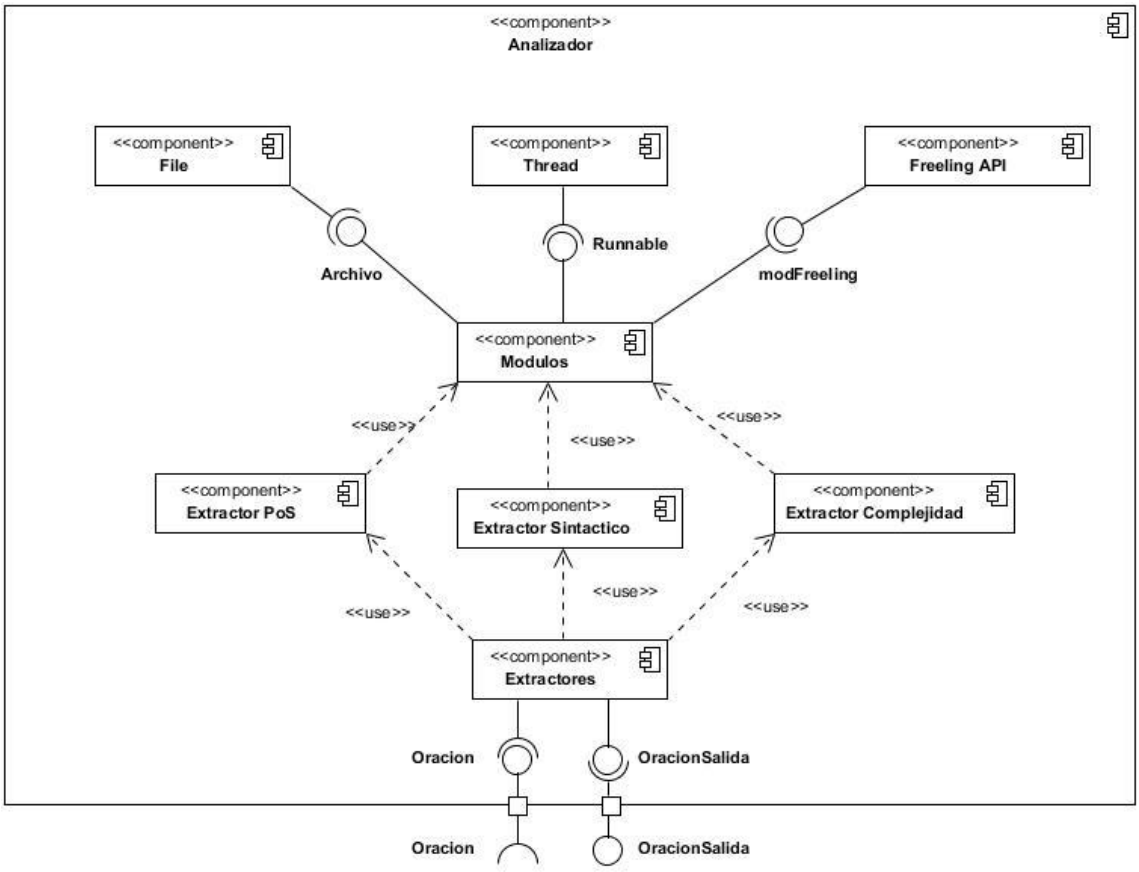

Figura 4.1. Diagrama de componentes del sistema de extractores de características.
Fuente: propia.

1.3 Pruebas unitarias de las funciones de extracción

Para probar el correcto funcionamiento de las funciones de extracción, se llevaron a cabo pruebas unitarias automatizadas con JUnit. Para cada función se definió un mínimo de cuatro casos de prueba (con sus resultados esperados) que debían ser pasados para que la función sea clasificada como correcta. Para ello, se tomaron oraciones en donde la característica estaba presenta 1 vez (caso trivial), cuando la característica estaba presente más de 1 vez, cuando podía confundirse con una característica similar y cuando estaba ausente. Para más detalles, revisar el **plan de pruebas** en el anexo 8.

2 Selección de clasificadores y atributos

En esta sección se describirá el proceso estadístico seguido para la selección de los algoritmos de clasificación y subconjuntos de características que serán extraídos de los textos, el cual pudo ser iniciado una vez disponibles los extractores de características. Cabe mencionar que el proceso mencionado a continuación es genérico y fue repetido para cada operación de simplificación.

Para obtener los datos a ser utilizados en esta sección se hizo uso de las interfaces de exploración y experimentación de Weka. En particular, se usó la interfaz de exploración de la herramienta para pre procesar los datos contenidos en los archivos de origen (de extensión .ARFF) y llevar a cabo la selección de atributos. Por otro lado, se usó su interfaz de experimentación para llevar a cabo los experimentos diseñados para seleccionar el algoritmo y el subconjunto de atributos con el que deberá trabajarse.

2.1 Pre-procesamiento de datos

En primer lugar, con la finalidad de reducir la variabilidad producida por las diferentes escalas de medición de los diferentes atributos extraídos, se llevó a cabo una normalización de los mismos. Es decir, todos los atributos fueron escalados para que sus valores estuvieran en el rango de 0 a 1. Esto se logró utilizando el filtro ***"Normalize"*** de Weka sobre los datos del archivo fuente (ARFF) en la interfaz del Explorador. Los resultados fueron guardados en nuevos archivos ARFF para su posterior uso.

Luego, para formar los subconjuntos de atributos con los cuales trabajar, se usó el método ***"CfsSubsetEval"*** (disponible en la interfaz de exploración de Weka), el cual es un método independiente del esquema de clasificación. Se trata de una característica importante ya que gracias a ella es justificable el uso del mismo conjunto de atributos con diferentes clasificadores. A diferencia de los métodos que sí dependen del esquema de clasificación usado, este escoge los atributos que se correlacionan fuertemente con la etiqueta de clase y débilmente con otros atributos.

El método requiere de un algoritmo de búsqueda para recorrer el espacio de soluciones y generar subconjuntos potenciales. Como se mencionó en la sección de métodos, se utilizó una búsqueda de dejar un elemento fuera por vez ("Leave one out"). Para ello, en Weka, se configuró un método de búsqueda llamado ***"Best First"*** de forma tal que haga un recorrido en sentido inverso por el espacio de soluciones; es decir, que empiece por soluciones que incluyan todos los atributos y

luego vaya avanzando a otras que tengan un atributo menos cada vez. Con los resultados obtenidos en este paso, diferentes para cada operación, se creó nuevos archivos ARFF que contenían solo los atributos seleccionados por el algoritmo.

2.2 Experimentación

Una vez pre procesado el conjunto de oraciones de entrenamiento, se procedió con la experimentación. Para ello, en la interfaz de experimentación de Weka se configuró un nuevo ensayo con los dos conjuntos de datos formados en la etapa anterior: datos con el conjunto completo de características y datos con un conjunto reducido de características. Asimismo, se decidió utilizar los siguientes algoritmos de clasificación:

- **ZeroR:** clasifica todas las instancias como la clase mayoritaria. Servirá como línea base.
- **Bayes Simple:** uno de los más simples que se puede encontrar y que asume la independencia condicional entre las características.
- **J48:** que es uno que genera un árbol de decisión.
- **SMO (implementación de una máquina de vector de soporte),** que es un clasificador de creciente uso en el campo debido a su menor necesidad de heurísticas para su entrenamiento.
- **Logístico:** o clasificador de máxima entropía. Frecuentemente usado en el campo cuando no es posible asumir la independencia condicional entre las características; es decir, cuando no se puede usar Bayes Simple.

Una vez configurado el experimento, se procedió a llevarlo a cabo. Se corrió 10 iteraciones de validaciones cruzadas de 10 dobleces ("*10-fold cross validation*"). En vista que en cada validación cruzada se ejecuta 10 veces el entrenamiento y prueba del clasificador, al final se tiene el equivalente a 100 puntos de datos por clasificador (10 iteraciones * 10 entrenamientos/iteración). No obstante, puesto que se trata de una comparación de medias, es posible trabajar sólo con las medias obtenidas en cada validación cruzada; es decir, 10 valores por clasificador. Esto se debe a una propiedad de la media:

$$\bar{\boldsymbol{X}} = \frac{1}{n}\sum_{i=k}^{n} X_i = \frac{1}{n}\sum_{i=1}^{n} X_i = \frac{1}{n}\sum_{i=1}^{k} X_i + \frac{1}{n}\sum_{i=k+1}^{n} X_i \ldots (\boldsymbol{Ec.\,4.\,1.})$$

Es decir, que la media global puede ser descompuesta en (n/k) partes de k-elementos cada una. Resulta lógico que en el caso de este trabajo, k = 10.

Luego, para evaluar los resultados; es decir, el desempeño de los distintos clasificadores en diferentes contextos se seleccionó la métrica "Medida F promedio". La medida F se define como la media harmónica de la precisión y la memoria (sección 2.4.3) según:

$$\boldsymbol{Medida\ F} = 2 * \frac{precision * memoria}{precision + memoria} \quad \ldots (\boldsymbol{Ec.\ 4.2.})$$

Por tanto, la "Medida F promedio" se define como el promedio de la medida F para la etiqueta "Sí" y la medida F para la etiqueta "No".

Se seleccionó esta métrica porque representa la combinación de dos de las medidas de evaluación de desempeño más importantes en el campo de estudio (clasificación de textos). Otro motivo para su uso constituye el hecho que es una medida encontrada frecuentemente en la literatura científica, lo que facilita la comparación de resultados. Los resultados fueron exportados a Excel y SPSS, herramientas que se usaron para el análisis de los mismos.

A nivel estadístico, los experimentos fueron diseñados con el fin de seleccionar al clasificador con mejor desempeño para cada operación y el mejor conjunto de atributos que funciona con el mismo. Para simplificar el estudio, se consideró solo dos posibles conjuntos de atributos: el formado por los 37 atributos mencionados en la sección anterior y el subconjunto de atributos seleccionado por un algoritmo específicamente diseñado para eso.

Para hacer las comparaciones entre las medias de los resultados obtenidos en cada caso (y así identificar al clasificador con mayor valor promedio de la medida F), se llevó a cabo una prueba de pares T ("T-paired simple test"). En esta prueba se tiene dos hipótesis:

- **Hipótesis nula:** la diferencia de las medias entre las dos poblaciones es cero.
- **Hipótesis alternativa:** la diferencia de las medias entre las dos poblaciones es distinta de cero.

Para poder llevarla a cabo, en cada caso, se requirió comprobar cuatro supuestos. Dichos supuestos son:

1. Se tiene una **variable dependiente** que sea medida en el rango de valores de tipo continuo. En este caso, la medida F es una métrica que cumple dicho requisito, pues pertenece al conjunto de los números reales.
2. Se tiene una **variable independiente** que tenga dos posibles grupos. En este caso, se tomó como variable la combinación de las variables "clasificador" y "conjunto de atributos". De esta forma, se obtuvo 10 posibles grupos, los cuales fueron etiquetados con el patrón: [ZeroR | Bayes | J48 | Logistico | SMO] _ [Completo | Reducido] (por ejemplo, Bayes_Completo). En vista que dicha variable solo puede tener dos posibles grupos, se repitió la prueba varias veces, cada vez probando pares diferentes (por ejemplo: Bayes_Completo vs Bayes_Reducido). Con la finalidad de reducir el número de comparaciones necesarias, primero se hizo la comparación entre el conjunto completo y reducido de atributos para cada clasificador. Finalmente, se procedió a comparar el desempeño de los distintos clasificadores con su mejor conjunto de atributos.
3. Que **no existan valores extremos** en las diferencias de las medidas. Esto se comprobó usando diagramas de cajas para cada prueba.
4. Que las diferencias de las medidas tengan una d**istribución aproximada a la normal**. Esto se probó en cada caso aplicando la prueba de Shapiro-Wilk. En esta prueba, la hipótesis nula consiste en que la distribución de los datos es normal y la hipótesis alternativa, en que los datos no tienen tal distribución.

Como se mencionó anteriormente, se usó como datos solo la media de cada validación cruzada (N=10). Hacer esto, facilitó los puntos (3) y (4), que son comprobados a nivel de diferencia entre medias y no entre datos individuales.

2.3 Resumen del método

Para un mejor entendimiento del método utilizado, se resume lo descrito en las dos secciones anteriores. El esquema de trabajo puede resumirse en los siguientes puntos:

1. Se extrajo las características de los textos del corpus usando los extractores de características. Las mismas fueron grabadas en archivos de extensión ARFF.
2. Se pre-procesaron dichos archivos en Weka. En particular, se les aplicó el filtro de normalización.

3. Se llevó a cabo la experimentación para seleccionar cuál es el mejor conjunto de atributos para cada clasificador. Es decir, se comparó los pares X_Completo y X_Reducido (donde X es el clasificador) Para comparar dichos desempeños, se utilizó la medida F promedio como variable dependiente en varias pruebas T para pares.
4. Se llevó a cabo la experimentación para seleccionar el mejor par clasificador-atributos para la operación. En este caso, se comparó los pares ZeroR_Y, Bayes_Y,J48_Y,Log_Y y SMO_Y (donde Y es el conjunto de atributos seleccionado en la experimentación del paso 3). Nuevamente, se utilizó la medida F promedio como variable dependiente en varias pruebas T para pares.

En la siguiente sección, se describirá detalladamente el proceso para la selección del clasificador y atributos para la operación de eliminación. El método es el mismo para las otras operaciones, cuyos casos no serán descritos con tanto nivel de detalle.

2.4 Eliminación

Una vez pre procesado los datos y configurado el algoritmo de selección de atributos como se describió anteriormente, se ejecutó la simulación usando validación cruzada estratificada de 10 dobleces (*"10-fold stratified cross-validation"*). De esta forma, en cada una de las 10 repeticiones, se seleccionaron los atributos sobre el subconjunto de entrenamiento y se probó su efectividad sobre el subconjunto de validación restante. Así se evitó que la evaluación del desempeño del clasificador con un subconjunto de características dado se llevara a cabo usando datos que fueron empleados en su entrenamiento. El resultado obtenido fue el que se observa en la tabla 4.2.

Tabla 4.2. Selección de atributos con Validación Cruzada

Atributo	Porcentaje
Posición	100%
Porcentaje de repetidas	100%
Tamaño de frase nominal	40%
Indicativos	20%

Fuente: propia

Los resultados de la tabla 4.2 se interpretan de la siguiente forma: el atributo X fue sido seleccionado el Y% de veces sobre el set de entrenamiento en la validación cruzada. Así, por ejemplo, en una validación cruzada de 10 repeticiones

(que es lo que se ha corrido), el atributo “Indicativos” fue seleccionado 2 veces (20%) como miembro del subconjunto de atributos. De los resultados obtenidos puede apreciarse claramente que los dos atributos más importantes fueron la posición y el porcentaje de repetidas, que fueron seleccionadas en todos los casos. Asimismo, tanto el tamaño de la frase nominal como el número de verbos en modo indicativo fueron seleccionados por lo menos una vez, por lo que también fueron considerados dentro del subconjunto plausible de características.

Una vez obtenido el subconjunto de características, se procedió a evaluar el desempeño de los clasificadores. Por ello, utilizando la interfaz de experimentación de Weka, se configuró el experimento como se describió anteriormente y se ejecutó. Los resultados obtenidos pueden verse resumidos en la figura 4.2, los cuales se usarán más adelante.

Descriptive Statistics

	Minimum	Maximum	Mean	Std. Deviation
ZeroR_Completo	,6555	,6555	,655500	0E-7
ZeroR_Reducido	,6555	,6555	,655500	0E-7
Bayes_Completo	,5158	,5290	,520710	,0043445
Bayes_Reducido	,7036	,7136	,708090	,0026223
J48_Completo	,6691	,6973	,682880	,0090624
J48_Reducido	,6534	,6607	,656910	,0024177
Logistico_Completo	,6997	,7124	,706100	,0039978
Logistico_Reducido	,6844	,6877	,686020	,0010465
SMO_Completo	,6723	,6947	,686800	,0065359
SMO_Reducido	,6523	,6621	,658720	,0032141

Figura 4.2. Estadísticas descriptivas de la ejecución del experimento.
Fuente: Propia, con apoyo de IBM SPSS 20.

Como se mencionó anteriormente, en un primer momento se hizo la comparación a nivel de conjunto de características para cada clasificador. Para ello, se tuvo que comprobar que para cada par se cumplieran los supuestos (3) y (4) de la prueba.

Para probar el supuesto 3, la ausencia de valores extremos, se usó un gráfico de diagramas de cajas. Como se aprecia en la figura 4.3, hay un caso donde la diferencia de medias tiene un valor extremo. Para superar este problema, se procedió a eliminar dicho valor; es decir, excluyendo las mediciones que originaron dicho valor del estudio.

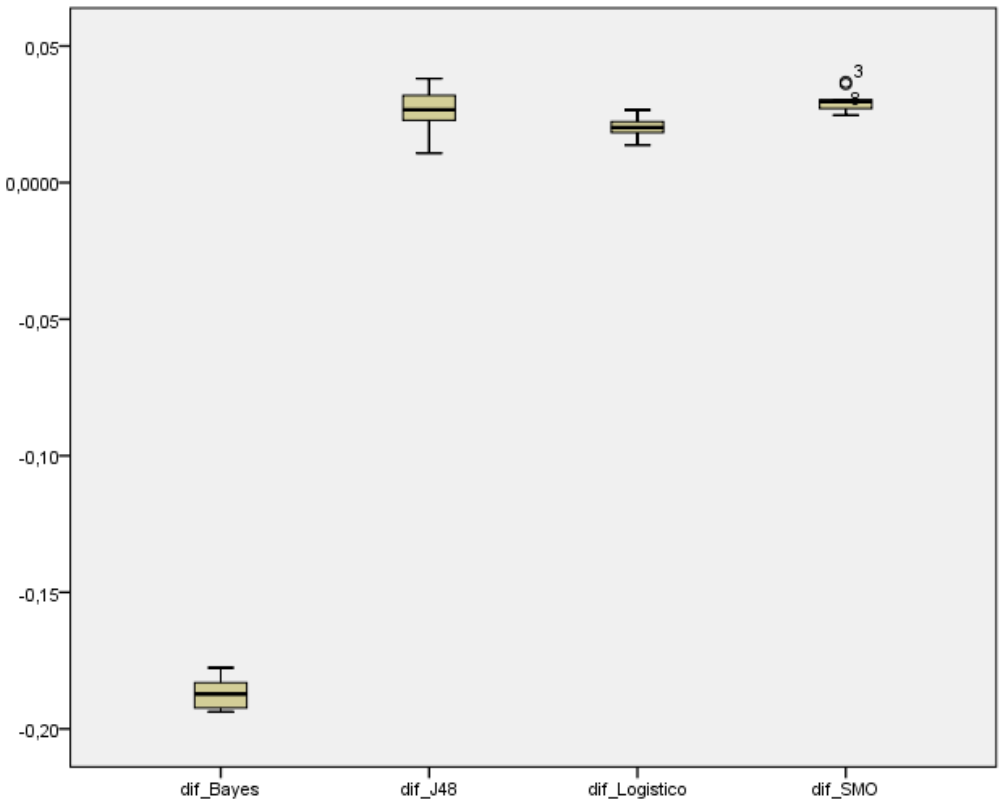

Fig. 4.3. Diagramas de cajas de diferencias de medias.
Fuente: Propia, con apoyo de IBM SPSS 20.

Para probar el supuesto 4, la normalidad de las diferencias de medias, se efectuó la prueba de Shapiro-Wilk. Tal como se ve en la figura 4.4, se cumple para todos los casos ya que siempre se cumple que sig (p) > 0.05 (nivel de aceptación de la prueba). Por ejemplo, para la fila dif_Bayes, que representa la diferencia entre la media de la medida F obtenida para el clasificador bayesiano con el conjunto de atributos completo y la media de la medida F obtenida para el mismo clasificador, pero con el conjunto de atributos reducido, el valor p es mayor que 0.05. Por tanto, se acepta la hipótesis nula El mismo criterio se aplicó para el resto de filas.

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_Bayes	,163	10	,200*	,939	10	,546
dif_J48	,159	10	,200*	,943	10	,588
dif_Logistico	,138	10	,200*	,947	10	,638
dif_SMO	,200	9	,200*	,861	9	,099

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Fig. 4.4. Resultados de las pruebas de normalidad (Shapiro-Wilk).
Fuente: Propia, con apoyo de IBM SPSS 20.

En vista que todas las suposiciones del modelo se cumplen, es posible efectuar la prueba de pares T. Como puede apreciarse en la figura 4.5., se observan diferencias significativas en cada par (sig<0.0005, por lo que se rechaza la hipótesis nula).

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 1	Bayes_Reducido - Bayes_Completo	,1873800	,0051697	,0016348	114,618	9	,000
Pair 2	J48_Reducido - J48_Completo	-,0259700	,0087846	,0027779	-9,349	9	,000
Pair 3	Logistico_Reducido - Logistico_Completo	-,0200800	,0042536	,0013451	-14,928	9	,000
Pair 4	SMO_Reducido - SMO_Completo	-,0280800	,0064529	,0020406	-13,761	9	,000

Figura 4.5. Resultado de la prueba de pares T,
Fuente: Propia, con apoyo de IBM SPSS 20.

A partir de lo observado en la columna “Mean” de la figura 4.4 y corroborando con las estadísticas descriptivas de la figura 4.2, puede deducirse que:

$$Bayes_{Completo} < Bayes_{Reducido} \quad \dots (\boldsymbol{a})$$

$$J48_{Reducido} < J48_{Completo} \quad \dots (\boldsymbol{b})$$

$$Logistico_{Reducido} < Logistico_{Completo} \quad \dots (\boldsymbol{c})$$

$$SMO_{Reducido} < SMO_{Completo} \quad \dots (\boldsymbol{d})$$

En consecuencia, se utilizaría el set reducido de atributos (seleccionado) con el clasificador de Bayes y se usaría el conjunto completo con los otros clasificadores.

Para ZeroR no se hizo el análisis debido a que por su naturaleza, es indistinto que set de atributos se utilice.

Como se describió en la sección anterior, una vez seleccionado el conjunto de atributo, se llevará a cabo la misma prueba de pares para comparar el desempeño de los clasificadores. Por consiguiente, se procedió a calcular las diferencias entre las posibles combinaciones de medias de los clasificadores: ZeroR_Completo, Bayes_Reducido, J48_Completo, Logistico_Completo y SMO_Completo.

Nuevamente, se revisó la existencia de valores extremos. Como se evidencia en la figura 4.6, en este caso sí se observan tales casos en los datos. Por ello, se procedió a eliminar las medidas del caso para excluirlas del análisis.

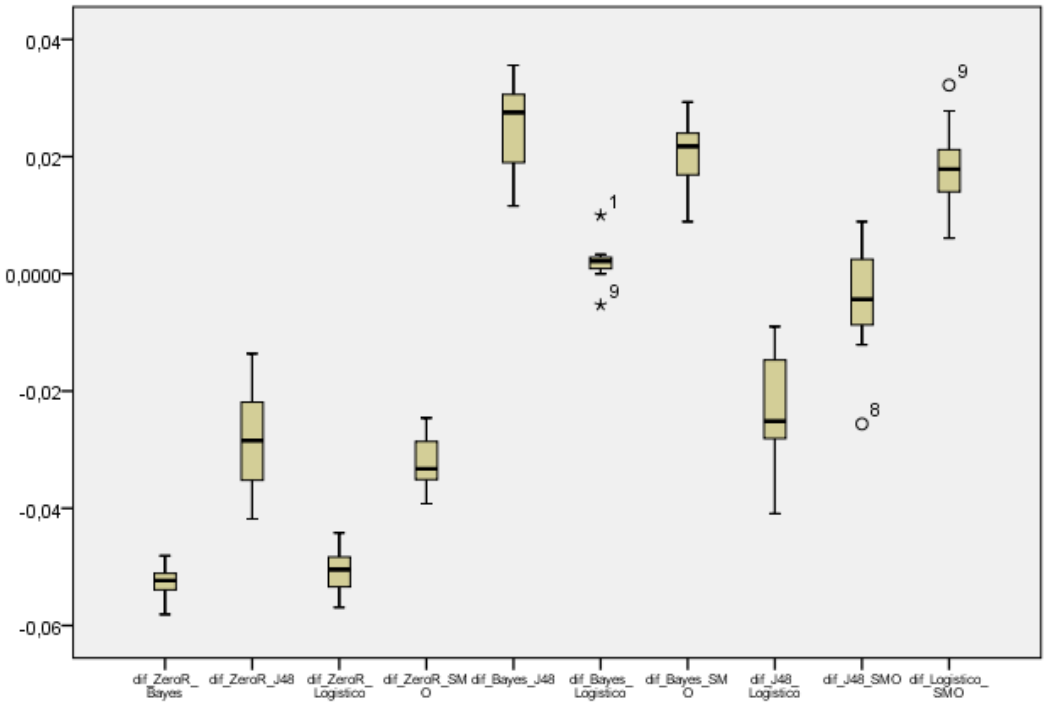

Fig. 4.6. Diagrama de cajas para encontrar valores atípicos y extremos.
Fuente: Propia, con apoyo de IBM SPSS 20.

Cabe mencionar que, como los valores extremos correspondían a campos que fueron calculados como la diferencia de dos medias, se tuvo que eliminar el valor y los dos datos a partir de los cuales se calculó dicha medida.

Enseguida, se procedió a comprobar la normalidad de las diferencias de las medias usando la prueba de Shapiro-Wilk. Como se aprecia en la figura 4.7, se cumple la distribución normal en todos los casos ya que sig > 0.05 (nivel de significancia de la prueba).

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_ZeroR_Bayes	,236	7	,200*	,850	7	,122
dif_ZeroR_J48	,195	7	,200*	,931	7	,558
dif_ZeroR_Logistico	,156	7	,200*	,971	7	,903
dif_ZeroR_SMO	,276	7	,115	,896	7	,310
dif_Bayes_J48	,256	7	,184	,885	7	,252
dif_Bayes_Logistico	,230	7	,200*	,929	7	,546
dif_Bayes_SMO	,183	7	,200*	,950	7	,731
dif_J48_Logistico	,271	7	,129	,877	7	,214
dif_J48_SMO	,131	7	,200*	,986	7	,985
dif_Logistico_SMO	,190	7	,200*	,914	7	,421

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figura 4.7. Resultados de la prueba de normalidad de Shapiro-Wilk.

Fuente: Propia, con apoyo de IBM SPSS 20.

Finalmente, como se cumplen todos los supuestos, se efectuó la prueba de pares.Se obtuvo los resultados que se muestran en la figura 4.8. Analizando los resultados como se hizo en la prueba anterior, en base al valor p (o columna “Sig”), se puede apreciar que sólo en el caso de J48 y SMO p>0.05, por lo que sólo en ese caso se acepta la hipótesis nula de que ambos clasificadores tienen el mismo desempeño (equivalentemente, la diferencia de medias es cero). En los otros casos, se puede concluir que existe una diferencia significativa entre el desempeño de los algoritmos. En base a los resultados de la figura 4.8, que también indica si la diferencia de las medias es positiva o negativa, se puede concluir que:

$$ZeroR < Bayes, \quad ZeroR < J48, \quad ZeroR < Logistico, \quad ZerpR < SMO \ldots (\boldsymbol{a})$$

$$Bayes > J48, \quad Bayes > Logistico, \quad Bayes > SMO \ldots (\boldsymbol{b})$$

$$J48 < Logistico, \quad J48 = SMO \ldots (\boldsymbol{c})$$

$$Logistico > SMO \ldots (\boldsymbol{d})$$

Entonces, de (a), (b), (c) y (d):

$$\boldsymbol{ZeroR < J48 = SMO < Logistico < Bayes}$$

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 1	ZeroR_Completo - Bayes_Reducido	-,0523143	,0030569	,0011554	-45,278	6	,000
Pair 2	ZeroR_Completo - J48_Completo	-,0273800	,0090624	,0028658	-9,554	9	,000
Pair 3	ZeroR_Completo - Logistico_Completo	-,0503571	,0031501	,0011906	-42,295	6	,000
Pair 4	ZeroR_Completo - SMO_Completo	-,0313000	,0065359	,0020668	-15,144	9	,000
Pair 5	Bayes_Reducido - J48_Completo	,0243857	,0084347	,0031880	7,649	6	,000
Pair 6	Bayes_Reducido - Logistico_Completo	,0019571	,0012660	,0004785	4,090	6	,006
Pair 7	Bayes_Reducido - SMO_Completo	,0179857	,0051628	,0019514	9,217	6	,000
Pair 8	J48_Completo - Logistico_Completo	-,0224286	,0076301	,0028839	-7,777	6	,000
Pair 9	J48_Completo - SMO_Completo	-,0039200	,0105872	,0033480	-1,171	9	,272
Pair 10	Logistico_Completo - SMO_Completo	,0160286	,0052389	,0019801	8,095	6	,000

Fig. 4.8. Resultado de la comparación de pares T.

Fuente: Propia, con apoyo de IBM SPSS 20.

Por tanto, combinando lo anterior con el primer resultado, se llegó a la conclusión que para la operación de eliminación debe utilizarse a “Naive Bayes” como clasificador. Adicionalmente, que este algoritmo se desempeña mejor con el conjunto reducido de atributos.

2.5 Separación

La segunda operación a analizar, por su importancia y frecuencia, es la separación. Como se mencionó inicialmente, se explicó detalladamente el procedimiento seguido para la operación de eliminación. Puesto que el proceso es similar para esta operación, se explicará más resumidamente los resultados obtenidos.

La selección de atributos se configuró de la misma forma que se describió anteriormente. Los resultados obtenidos se resumen en la tabla 4.3.

Tabla 4.3. Características seleccionadas por el algoritmo “CfsSubsetEval”.

#	Característica	%	#	Característica	%
1	Conjunciones coordinativas	100%	12	Sujeto primero	100%
2	Conjunciones subordinativas	100%	13	Porcentaje de repetidas	100%
3	Tamaño frase nominal	100%	14	Subjuntivo	80%
4	Tamaño frase preposicional	100%	15	Número de cláusulas relativas	80%
5	Num. de conjunciones coordinadas	100%	16	Es oración compuesta	40%
6	Num. de conjunciones subordinadas	100%	17	Tamaño de frase verbal	20%
7	Número de cláusulas coordinadas	100%	18	Número de palabras por oración	20%
8	Número de frases nominales	100%	19	Número de frases preposicionales	10%
9	Tamaño promedio de palabras	100%	20	Número de sustantivos	10%
10	Numero de caracteres en oración	100%	21	Número de sustantivos propios	10%
11	Posición	100%	22	Número de infinitivos	10%

Fuente: propia.

De manera similar a como se procedió con la operación de eliminación. Se configuró un experimento en el experimentador de Weka, de forma similar a lo que se describió previamente. Los resultados obtenidos se muestran en la figura 4.9, que será usada más adelante.

Luego, se efectuó una prueba de pares T para seleccionar el mejor conjunto de características para cada clasificador. Para ello, primero se verificó con diagramas de cajas si existían valores extremos en las diferencias de medias. En vista de que tales valores no existían, no fue necesario eliminar ningún punto de dato del estudio.

Descriptive Statistics

	Minimum	Maximum	Mean	Std. Deviation
ZeroR_Completo	,5372	,5372	,537200	0E-7
ZeroR_Reducido	,5372	,5372	,537200	0E-7
Bayes_Completo	,6070	,6147	,610680	,0025728
Bayes_Reducido	,5956	,6019	,598500	,0020526
J48_Completo	,6702	,7162	,693580	,0125314
J48_Reducido	,6939	,7206	,711010	,0076314
Logistico_Completo	,7142	,7268	,720720	,0041378
Logistico_Reducido	,7227	,7333	,727600	,0033625
SMO_Completo	,7142	,7282	,722270	,0045903
SMO_Reducido	,7082	,7251	,717470	,0053862

Figura 4.9. Estadísticas descriptivas del resultado del experimento.
Fuente: propia, con apoyo de IBM SPSS 20.

Asimismo, se probó todas las diferencias seguían una distribución normal usando la prueba de Shapiro-Wilk. En dicha prueba, se observó que p>0.05 para

todos los casos, por lo que se aceptó la hipótesis nula de la misma en todos los casos. Para evidenciar la afirmación, se muestra la tabla de resultados de la prueba en la figura 4.10.

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_Bayes	,181	10	,200*	,900	10	,217
dif_J48	,237	10	,117	,899	10	,212
dif_Log	,217	10	,200*	,867	10	,093
dif_SMO	,183	10	,200*	,946	10	,616

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figura 4.10. Resultados de la prueba de normalidad de Shapiro-Wilk.
Fuente: propia, con apoyo de IBM SPSS 20.

Una vez comprobadas las suposiciones, se ejecutó la prueba de pares T. Sus resultados se muestran en la figura 4.11.

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 2	Bayes_Completo - Bayes_Reducido	,0121800	,0015648	,0004948	24,615	9	,000
Pair 3	J48_Completo - J48_Reducido	-,0174300	,0121123	,0038303	-4,551	9	,001
Pair 4	Logistico_Completo - Logistico_Reducido	-,0068800	,0047884	,0015142	-4,544	9	,001
Pair 5	SMO_Completo - SMO_Reducido	,0048000	,0067295	,0021281	2,256	9	,051

Figura 4.11. Resultado de la prueba de pares T.
Fuente: propia, con apoyo de IBM SPSS 20.

Por el valor de la columna “Sig”, puede concluirse que en casi todos los casos la hipótesis nula es rechazada; es decir, sí existe una diferencia significativa entre las medias obtenidas. La única excepción ocurre con el SMO, para el cual su desempeño con cualquier conjunto de características es estadísticamente el mismo. Adicionalmente, a partir de los resultados de la figura, puede concluirse que:

$$Bayes_{Completo} > Bayes_{Reducido} \quad \dots (\boldsymbol{a})$$

$$J48_{Completo} < J48_{Reducido} \quad \dots (\boldsymbol{b})$$

$$Logistico_{Completo} < Logistico_{Reducido} \quad \dots (\boldsymbol{c})$$

$$SMO_{Completo} = SMO_{Reducido} \quad \ldots . (\boldsymbol{d})$$

En vista que para el SMO y ZeroR, por diferentes motivos, es indistinto el conjunto de atributos que se selecciones para trabajar con ellos, se considerará que se trabaja con el set completo de atributos.

Para comparar las medidas de desempeño de los algoritmos, como en el caso de eliminación, se efectuó una prueba de pares. Para ello, con la ayuda de diagramas de cajas se descartó la existencia de valores extremos en la diferencia de medias; asimismo, con la prueba de Shapiro-Wilk se comprobó el ajuste a la normal de dichas diferencia. Los resultados de esta última prueba se muestran en la figura 4.12.

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_ZeroR_Bayes	,188	10	,200*	,945	10	,605
dif_ZeroR_J48	,186	10	,200*	,922	10	,373
dif_ZeroR_Log	,161	10	,200*	,960	10	,782
dif_ZeroR_SMO	,216	10	,200*	,910	10	,281
dif_Bayes_J48	,238	10	,115	,904	10	,243
dif_Bayes_Log	,168	10	,200*	,936	10	,507
dif_Bayes_SMO	,167	10	,200*	,887	10	,158
dif_J48_Log	,253	10	,070	,892	10	,178
dif_J48_SMO	,146	10	,200*	,912	10	,292
dif_Log_SMO	,094	10	,200*	,992	10	,998

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figura 4.12. Resultados de la prueba de normalidad de Shapiro-Wilk.
Fuente: propia, con apoyo de IBM SPSS 20.

Comprobadas las suposiciones, se llevó a cabo la prueba T. Los resultados de la misma se muestran en la figura 4.13. En ella puede apreciarse que en todos los casos se rechaza la hipótesis nula; es decir, que la diferencia de las medias en cada par es estadísticamente significativa. Esto, en combinación con la información acerca del signo de la diferencia (y con la información de la figura 4.9 como referencia adicional), permitió concluir que:

$$ZeroR < Bayes, \quad ZeroR < J48, \quad ZeroR < Logistico, \quad ZeroR < SMO \ldots (\boldsymbol{a})$$

$$Bayes < J48, \quad Bayes < Logistico, \quad Bayes < SMO \qquad \ldots (\boldsymbol{b})$$

$$J48 < Logistico, \qquad J48 < SMO \qquad \ldots (\boldsymbol{c})$$

$$Logistico > SMO \qquad \ldots (\boldsymbol{d})$$

Combinando (a), (b), (c) y (d), se obtiene que:

$$\boldsymbol{ZeroR < Bayes < J48 < SMO < Logistico}$$

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 1	ZeroR_Completo - Bayes_Completo	-,0734800	,0025728	,0008136	-90,314	9	,000
Pair 2	ZeroR_Completo - J48_Reducido	-,1738100	,0076314	,0024133	-72,023	9	,000
Pair 3	ZeroR_Completo - Logistico_Reducido	-,1904000	,0033625	,0010633	-179,060	9	,000
Pair 4	ZeroR_Completo - SMO_Completo	-,1850700	,0045903	,0014516	-127,494	9	,000
Pair 5	Bayes_Completo - J48_Reducido	-,1003300	,0065238	,0020630	-48,633	9	,000
Pair 6	Bayes_Completo - Logistico_Reducido	-,1169200	,0053702	,0016982	-68,849	9	,000
Pair 7	Bayes_Completo - SMO_Completo	-,1115900	,0049552	,0015670	-71,213	9	,000
Pair 8	J48_Reducido - Logistico_Reducido	-,0165900	,0091475	,0028927	-5,735	9	,000
Pair 9	J48_Reducido - SMO_Completo	-,0112600	,0091382	,0028898	-3,897	9	,004
Pair 10	Logistico_Reducido - SMO_Completo	,0053300	,0069189	,0021879	2,436	9	,038

Figura 4.13. Resultado de la prueba de pares T.
Fuente: propia, con apoyo de IBM SPSS 20.

Por tal motivo, a partir de los resultados obtenidos con ambas pruebas, fue posible concluir que para la operación de separación, el clasificador más conveniente de usar es el logístico empleando el conjunto reducido de características.

2.6 Reducción

Para la reducción se procedió de manera similar a las anteriores operaciones. Luego de aplicar el filtro de normalización, se configuró y ejecutó el algoritmo para seleccionar el conjunto de atributos a ser utilizado por los clasificadores. Las características que forman parte del conjunto reducido son todas aquellas que se muestran en la tabla 4.4.

Tabla 4.4. Resultados de la selección de atributos

#	Atributos	%	#	Atributos	%
1	Núm. Caracteres por oración	100%	12	Num. Frases adverbiales	60%
2	Num. De palabras por oración	100%	13	Tam. Frases preposicionales	50%
3	Posición	100%	14	Indicativos	40%
4	Sujeto primero	100%	15	Participio	10%
5	Verbos	90%	16	Num. Frases verbales	10%
6	Porcentaje repetidas	80%	17	Profundidad	10%
7	Num. Frases nominales	60%			

Fuente: propia.

Al igual que con las otras operaciones, se configuró el experimento en la interfaz correspondiente de Weka, se recopiló los resultados y se analizaron con la ayuda del SPSS. Dichos resultados pueden observarse en la figura 4.14, la cual será utilizada más adelante.

Descriptive Statistics

	Minimum	Maximum	Mean	Std. Deviation
ZeroR_Completo	,6655	,6655	,665500	0E-7
ZeroR_Reducido	,6655	,6655	,665500	0E-7
Bayes_Completo	,5305	,5378	,534020	,0021280
Bayes_Reducido	,5583	,5677	,561900	,0031081
J48_Completo	,6599	,6866	,668220	,0081636
J48_Reducido	,6641	,6693	,666020	,0013497
Logistico_Completo	,6656	,6850	,680130	,0058082
Logistico_Reducido	,6778	,6871	,682360	,0024432
SMO_Completo	,6806	,7079	,692970	,0091264
SMO_Reducido	,6664	,6729	,669930	,0022598

Figura 4.14. Estadísticas descriptivas del resultado del experimento
Fuente: propia, con apoyo de IBM SPSS 20.

Luego, para la selección del conjunto de atributos a escogerse para cada clasificador, se procedió a llevar a cabo la prueba T de pares. Para ello, se comprobó el supuesto 3, la ausencia de valores extremos, usando diagramas de cajas. En vista de que existía un valor extremo, y este provenía de un dato calculado como diferencia de medias, se tuvo que eliminar dicho valor así como los dos puntos de datos que estuvieron involucrados en el cálculo de dicho valor.

Enseguida, se comprobó el ajuste a la distribución normal de las diferencias. En base a la prueba de Shapiro-Wilk, pudo verificarse que en todos los casos las diferencias se distribuyen normalmente (p>0.05 en todos los casos, tal como puede observarse en la figura 4.15).

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_Bayes	,157	10	,200*	,982	10	,975
dif_J48	,145	10	,200*	,945	10	,612
dif_Log	,161	8	,200*	,959	8	,798
dif_SMO	,218	10	,194	,926	10	,409

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figura 4.15. Resultados de la prueba de normalidad de Shapiro-Wilk.
Fuente: propia, con apoyo de IBM SPSS 20.

Comprobadas las suposiciones, se ejecutó la prueba de pares. Los resultados (mostrados en la figura 4.16), en combinación con la información de la figura 4.14, permitieron concluir qué conjunto de atributos era conveniente usar para cada clasificador.

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 1	Bayes_Completo - Bayes_Reducido	-,0278800	,0029495	,0009327	-29,891	9	,000
Pair 2	J48_Completo - J48_Reducido	,0022000	,0088857	,0028099	,783	9	,454
Pair 3	Logistico_Completo - Logistico_Reducido	,0003375	,0020770	,0007343	,460	7	,660
Pair 4	SMO_Completo - SMO_Reducido	,0230400	,0091070	,0028799	8,000	9	,000

Figura 4.16. Resultado de la prueba de pares T.
Fuente: propia, con apoyo de IBM SPSS 20.

Para los clasificadores de Bayes y SMO, se rechazó la hipótesis nula, pues p<0.0005 para ambos; es decir, la diferencia de las dos medias es significativa. En los casos de los otros dos clasificadores (J48 y Logístico), por el contrario, se tuvo que aceptar la hipótesis nula (p>0.05). Es decir, para J48 y Logístico, no existe una diferencia significativa entre aplicarlos con el conjunto completo de atributos o el seleccionado por el algoritmo “CfsSubsetEval”. En consecuencia:

$$Bayes_{Completo} < Bayes_{Reducido} \quad \dots (\boldsymbol{a})$$

$$J48_{Completo} = J48_{Reducido} \quad \dots (\boldsymbol{b})$$

$$Logistico_{Completo} = Logistico_{Reducido} \quad \dots (\boldsymbol{c})$$

$$SMO_{Completo} > SMO_{Reducido} \quad \dots . (\boldsymbol{d})$$

Una vez escogido el subconjunto de características, se procedió a la selección del clasificador más apropiado para esta operación. Para ello, se comparó el desempeño de los clasificadores ZeroR (completo), Bayes (reducido), J48 (completo), Logístico (completo) y SMO (completo), por medio del uso de la prueba de pares nuevamente.

Para proceder con dicha prueba, se comprobó la ausencia de valores extremos con un diagrama de cajas y también el ajuste a la distribución normal con la prueba de Shapiro-Wilk, cuyos resultados se muestran en la figura 4.17. En la figura se puede comprobar que para todos los casos p>0.05. Por tanto, se acepta la hipótesis nula de la prueba para todos los casos; es decir, que existe una distribución normal en los datos.

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_ZeroR_Bayes	,151	10	,200*	,931	10	,459
dif_ZeroR_J48	,180	10	,200*	,883	10	,141
dif_ZeroR_Log	,227	8	,200*	,843	8	,081
dif_ZeroR_SMO	,124	10	,200*	,962	10	,807
dif_Bayes_J48	,205	10	,200*	,918	10	,344
dif_Bayes_Log	,186	8	,200*	,907	8	,333
dif_Bayes_SMO	,146	10	,200*	,969	10	,878
dif_J48_Log	,187	8	,200*	,913	8	,375
dif_J48_SMO	,137	10	,200*	,932	10	,463
dif_Log_SMO	,148	8	,200*	,956	8	,767

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figura 4.17. Resultados de la prueba de normalidad de Shapiro-Wilk.
Fuente: propia, con apoyo de IBM SPSS 20.

Con ambos supuestos comprobados, se ejecutó la prueba y se obtuvo los resultados que se muestran en la figura 4.18. Debido a que sig < 0.0005 para todos los casos, excepto para el par 2, se puede afirmar que existen diferencias significativas entre las medidas de desempeño de los clasificadores involucrados en cada par. Por el contrario, para el par 2, se puede afirmar que el desempeño de los

clasificadores J48 y ZeroR es estadísticamente el mismo. La información mostrada en la figura 4.18, en conjunto con la información sobre las medias de la figura 4.14, permite concluir que:

$$ZeroR > Bayes, \quad ZeroR = J48, \quad ZeroR < Logistico, \quad ZerpR < SMO \quad \dots (\boldsymbol{a})$$

$$Bayes < J48, \quad Bayes < Logistico, \quad Bayes < SMO \quad \dots (\boldsymbol{b})$$

$$J48 < Logistico, \quad J48 < SMO \quad \dots (\boldsymbol{c})$$

$$Logistico < SMO \quad \dots (\boldsymbol{d})$$

Combinando (a), (b), (c) y (d), se obtiene que:

$$\boldsymbol{Bayes < ZeroR = J48 < Logistico < SMO}$$

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 1	ZeroR_Completo - Bayes_Reducido	,1036000	,0031081	,0009829	105,407	9	,000
Pair 2	ZeroR_Completo - J48_Completo	-,0027200	,0081636	,0025815	-1,054	9	,320
Pair 3	ZeroR_Completo - Logistico_Completo	-,0166625	,0028400	,0010041	-16,595	7	,000
Pair 4	ZeroR_Completo - SMO_Completo	-,0274700	,0091264	,0028860	-9,518	9	,000
Pair 5	Bayes_Reducido - J48_Completo	-,1063200	,0099428	,0031442	-33,815	9	,000
Pair 6	Bayes_Reducido - Logistico_Completo	-,1208000	,0030613	,0010823	-111,612	7	,000
Pair 7	Bayes_Reducido - SMO_Completo	-,1310700	,0106063	,0033540	-39,079	9	,000
Pair 8	J48_Completo - Logistico_Completo	-,0139625	,0095287	,0033689	-4,145	7	,004
Pair 9	J48_Completo - SMO_Completo	-,0247500	,0098414	,0031121	-7,953	9	,000
Pair 10	Logistico_Completo - SMO_Completo	-,0116250	,0105329	,0037239	-3,122	7	,017

Figura 4.18. Resultado de la prueba de pares T.
Fuente: propia, con apoyo de IBM SPSS 20.

Por tanto, combinando los dos grupos de pruebas de pares efectuados, puede concluirse que para esta operación deberá usarse un clasificador basado en SMO y que utilice el conjunto completo de características extraídas.

2.7 Cambio

Para la operación de cambio se procedió de manera similar a las anteriores operaciones. No obstante, debido a la distribución desigual de las clases (pocos

positivos en comparación con los negativos), fue necesario utilizar un filtro supervisado llamado ***"SMOTE"*** (Synthetic Minority Oversampling Technique). Este filtro utiliza el algoritmo de los k-vecinos más cercanos para interpolar los valores de los atributos y crear entidades sintéticas con una etiqueta equivalente a la de la clase minoritaria. En este caso, se incrementó el número de casos con etiqueta Sí en un 200%.

Luego de aplicar también el filtro de normalización, se ejecutó el algoritmo para la selección de atributos. De acuerdo a este último, los atributos que deben ser usados son los que se muestran en la tabla 4.5.

Tabla 4.5. Resultados de la selección de atributos

#	Atributo	%	#	Atributo	%
1	Conjunciones subordinadas	100%	9	Sustantivos propios	80%
2	Infinitivos	100%	10	Conjunciones coordinadas	70%
3	Participio	100%	11	Pronombre	70%
4	Subjuntivo	100%	12	Verbos	40%
5	Num. Frases adverbiales	100%	13	Num. Clausulas relativas	40%
6	Posicion	100%	14	Adverbios	20%
7	Sujeto primero	100%	15	Gerundios	10%
8	Indicativos	90%	16	Voz pasiva	10%

Fuente: propia

Enseguida, se ejecutó el experimento en la interfaz de experimentación de Weka. Los resultados obtenidos, que serán usados más adelante, se resumen en la figura 4.19. Después, para proceder con la prueba de pares para seleccionar el conjunto de atributos a ser usado con cada clasificador, se comprobó que los supuestos de dicha prueba se cumplan. Para ello, se calcularon las diferencias entre cada par (Clasificador_Completo, Clasificador_Reducido).

Con un diagrama de cajas, se comprobó la ausencia de valores extremos en las diferencias de medias. Asimismo, con la prueba de Shapiro-Wilk, se verificó que todas estas diferencias siguen una distribución normal.

Descriptive Statistics

	Minimum	Maximum	Mean	Std. Deviation
ZeroR_Completo	,6234	,6234	,623400	0E-7
ZeroR_Reducido	,6234	,6234	,623400	0E-7
Bayes_Completo	,4906	,4995	,495350	,0029194
Bayes_Reducido	,6553	,6630	,659340	,0022431
J48_Completo	,7657	,7923	,780560	,0097486
J48_Reducido	,7517	,7729	,761210	,0077292
Logistico_Completo	,6646	,6783	,673090	,0049355
Logistico_Reducido	,6431	,6549	,649180	,0040124
SMO_Completo	,8374	,8521	,844540	,0050344
SMO_Reducido	,7679	,7869	,775790	,0060940

Figura 4.19. Estadísticas descriptivas del resultado del experimento

Fuente: propia, con apoyo de IBM SPSS 20.

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_Bayes	,139	10	,200*	,930	10	,445
dif_J48	,196	10	,200*	,911	10	,290
dif_Log	,141	10	,200*	,926	10	,410
dif_SMO	,123	10	,200*	,957	10	,750

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figura 4.20. Resultados de la prueba de normalidad de Shapiro-Wilk.

Fuente: propia, con apoyo de IBM SPSS 20.

Comprobados los supuestos, se procedió a ejecutar la prueba de pares. A partir de los resultados de la figura 4.21, puede apreciarse que en todos los casos se rechaza la hipótesis nula ($p<0.0005$); es decir, en todos los casos existen diferencias significativas en el desempeño del clasificador con distintos conjuntos de atributos. Los resultados de esta prueba, en combinación con las estadísticas descriptivas, permitieron concluir que:

$$Bayes_{Completo} < Bayes_{Reducido} \quad \dots (\boldsymbol{a})$$

$$J48_{Completo} > J48_{Reducido} \quad \dots (\boldsymbol{b})$$

$$Logistico_{Completo} > Logistico_{Reducido} \quad \dots (\boldsymbol{c})$$

$$SMO_{Completo} > SMO_{Reducido} \quad \dots . (\boldsymbol{d})$$

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 1	Bayes_Completo - Bayes_Reducido	-,1639900	,0031043	,0009817	-167,054	9	,000
Pair 2	J48_Completo - J48_Reducido	,0193500	,0125667	,0039739	4,869	9	,001
Pair 3	Logistico_Completo - Logistico_Reducido	,0239100	,0061120	,0019328	12,371	9	,000
Pair 4	SMO_Completo - SMO_Reducido	,0687500	,0085022	,0026886	25,571	9	,000

Figura 4.21. Resultado de la prueba de pares T.

Fuente: propia, con apoyo de IBM SPSS 20.

Ya conocidos los conjuntos de datos que se deben utilizar con cada clasificador, se procedió a comparar el desempeño de los diferentes clasificadores. Nuevamente, se efectuaron pruebas de pares para ello. Por tal motivo, fue necesario comprobar los supuestos para todos los nuevos casos.

Nuevamente, un diagrama de cajas permitió comprobar que se cumple el supuesto referido a la ausencia de valores extremos. Por otra parte, la prueba de Shapiro-Wilk permitió corroborar que en todos los pares, las diferencias entre las medias sigue una distribución normal debido a que sig > 0.05 para todos los casos, tal como puede apreciarse en la figura 4.22.

Una vez comprobados ambos supuestos, se llevó a cabo la prueba de pares. Los resultados obtenidos (figura 4.23) muestran que en todos los casos existe una diferencia significativa entre las medias de cada par (sig < 0.0005 en todas las filas). La información de esta figura, combinada con la provista por la figura 4.19, permitió concluir que:

$$ZeroR < Bayes, \quad ZeroR < J48, \quad ZeroR < Logistico, \quad ZerpR < SMO \quad \dots (\boldsymbol{a})$$

$$Bayes < J48, \quad Bayes < Logistico, \quad Bayes < SMO \quad \dots (\boldsymbol{b})$$

$$J48 > Logistico, \quad J48 < SMO \quad \dots (\boldsymbol{c})$$

$$Logistico < SMO \quad \dots (\boldsymbol{d})$$

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
dif_ZeroR_Bayes	,129	10	,200*	,976	10	,942
dif_ZeroR_J48	,165	10	,200*	,915	10	,315
dif_ZeroR_Log	,229	10	,146	,866	10	,090
dif_ZeroR_SMO	,164	10	,200*	,944	10	,595
dif_Bayes_J48	,130	10	,200*	,953	10	,706
dif_Bayes_Log	,152	10	,200*	,950	10	,667
dif_Bayes_SMO	,109	10	,200*	,959	10	,779
dif_J48_Log	,227	10	,153	,914	10	,309
dif_J48_SMO	,202	10	,200*	,931	10	,456
dif_Log_SMO	,201	10	,200*	,900	10	,221

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figura 4.22. Resultados de la prueba de normalidad de Shapiro-Wilk.

Fuente: propia, con apoyo de IBM SPSS 20.

Paired Samples Test

		Paired Differences			t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean			
Pair 1	ZeroR_Completo - Bayes_Reducido	-,0359400	,0022431	,0007093	-50,667	9	,000
Pair 2	ZeroR_Completo - J48_Completo	-,1571600	,0097486	,0030828	-50,980	9	,000
Pair 3	ZeroR_Completo - Logistico_Completo	-,0496900	,0049355	,0015607	-31,838	9	,000
Pair 4	ZeroR_Completo - SMO_Completo	-,2211400	,0050344	,0015920	-138,906	9	,000
Pair 5	Bayes_Reducido - J48_Completo	-,1212200	,0105039	,0033216	-36,494	9	,000
Pair 6	Bayes_Reducido - Logistico_Completo	-,0137500	,0044891	,0014196	-9,686	9	,000
Pair 7	Bayes_Reducido - SMO_Completo	-,1852000	,0054770	,0017320	-106,929	9	,000
Pair 8	J48_Completo - Logistico_Completo	,1074700	,0104684	,0033104	32,464	9	,000
Pair 9	J48_Completo - SMO_Completo	-,0639800	,0084732	,0026795	-23,878	9	,000
Pair 10	Logistico_Completo - SMO_Completo	-,1714500	,0042610	,0013474	-127,241	9	,000

Figura 4.23. Resultado de la prueba de pares T.

Fuente: propia, con apoyo de IBM SPSS 20.

Combinando (a), (b), (c) y (d), se obtiene que:

$$\boldsymbol{ZeroR < Bayes < Logistico < J48 < SMO}$$

Adicionalmente, de ambos grupos de pruebas de pares, se puede concluir que el clasificador que debe ser utilizado para esta operación es SMO con el conjunto completo de atributos.

2.8 Inserción, Reordenamiento y Mantener

Como se indicó en el capítulo 3, la frecuencia de las operaciones de inserción y reordenamiento en el corpus es muy baja (menor al 2%). Como consecuencia, el conjunto de entrenamiento sufre de un desbalance muy fuerte entre instancias que tienen la etiqueta de clase “Sí” y las que tienen la etiqueta “No”. En particular, para la operación de reordenamiento, la relación “Si”/”No” es de 1 a 45; para la operación de inserción, es de 1 a 60.

Como consecuencia de este desbalance, un clasificador que prediga todo como la clase mayoritaria (en este caso, como un “No”), tendrá un equívoco buen desempeño. Este problema podría subsanarse usando un algoritmo que incremente la clase minoritaria, como “SMOTE”, tal como se hizo con la operación de cambio. No obstante, en estos casos la frecuencia de la operación es tan baja que el mencionado algoritmo tendría que crear artificialmente un 1000% más de instancias de la clase minoritaria. En este caso, tal enfoque no fue considerado adecuado debido a:

- El algoritmo tendría que crear más instancias con la etiqueta “Sí” usando los datos de aproximadamente 20 entidades base. Esto solo favorecería demasiado el sobreajuste porque generaría más instancias parecidas a las que ya están presentes en el corpus.
- Habrían más instancias artificiales que reales con la etiqueta “Si” en el corpus.

En vista a este desbalance, el cual ocasiona que los clasificadores tiendan a clasificar todo como “No”, no fue posible seleccionar un algoritmo de clasificación adecuado.

Por otra parte, la operación de Mantener no es otra cosa que “no hacer nada”. En consecuencia, no requiere de un clasificador específico: es el resultado de cuando los 4 clasificadores anteriormente entrenados dan como resultado una etiqueta “No” para una instancia determinada.

2.9 Conclusiones del capítulo

A lo largo del presente capítulo se ha descrito detalladamente el procedimiento seguido para poder obtener un modelo de clasificadores adecuado para tomar la decisión de llevar a cabo o no una operación de simplificación. Se empezó con la definición de 37 características de interés que se extraerían de los textos. Enseguida, se implementaron las funciones necesarias para la extracción de dichas características en el lenguaje de programación Java. Para conseguirlo, fue necesario adaptar y utilizar como soporte el paquete Freeling. Asimismo, se llevaron a cabo pruebas unitarias automatizadas con JUnit para asegurar el funcionamiento correcto de las funciones implementadas.

Adicionalmente, se acopló todo el componente analizador anteriormente descrito con la herramienta de anotación descrita en el capítulo 3, con la finalidad de proveerle la capacidad de poder convertir archivos de formato JSON en otros de formato ARFF, que es el que requiere Weka para leer sus datos de entrada.

Finalmente, se utilizó Weka para comparar, por cada operación, el desempeño de 5 clasificadores (ZeroR, Bayes, J48, Logístico y SMO), cada uno con dos posibles conjuntos de atributos: completos (los 37 mencionados anteriormente) y reducidos (seleccionados por un algoritmo independiente del esquema de clasificación). Así, para la operación de **"Eliminación"**, el clasificador con mejor desempeño fue un Bayesiano con el conjunto reducido de características; para la operación de **"Separación"**, un clasificador Logístico con el conjunto reducido de características; para la operación de **"Reducción"**, un clasificador de máquina de soporte vectorial (SMO) que utilice el conjunto completo de atributos; y, finalmente, para la operación de **"Cambio"**, otro clasificador SMO con el conjunto completo de atributos. Tal como se anticipó a partir del análisis del corpus, no fue posible seleccionar un clasificador para las operaciones de **"Reordenamiento"** e **"Inserción"**, las cuales son tan poco frecuentes que los clasificadores simplemente predicen todo como "No". Finalmente, la operación **"Mantener"** no requiere clasificador ya que su interpretación es simplemente "*no hacer nada*".

CAPITULO 5

1 Herramienta de Simplificación de Textos

Con la finalidad de hacer disponible la herramienta a las personas, se llevó a cabo la etapa de análisis, diseño e implementación de una aplicación web sencilla. Ello implicó la implementación de las operaciones de simplificación en el "backend" y la implementación de la vista de la aplicación en el "frontend".

1.1 Análisis

Los requerimientos para esta herramienta se encuentran registrados en la tabla 5.1.

Tabla 5.1. Catálogo de requerimientos.

#	Requerimiento	Prioridad	Tipo
1	La herramienta deberá poder recibir un texto completo y devolver como resultado el texto simplificado.	Alta	F
2	La herramienta deberá usar el clasificador Bayes Simple para decidir si llevar a cabo una operación de eliminación.	Alta	F
3	La herramienta deberá usar un clasificador Logístico para decidir si llevar a cabo una operación de separación.	Alta	F
4	La herramienta deberá usar un clasificador de máquina de soporte vectorial para decidir si llevar a cabo la operación de cambio.	Alta	F
5	La herramienta deberá usar un clasificador de máquina de soporte vectorial para decidir si llevar a cabo la operación de reducción.	Alta	F
6	La herramienta deberá efectuar la operación de simplificación correspondiente, si el clasificador así lo decide.	Alta	F
7	La herramienta será implementada usando el patrón MVC.	-	NF
8	La herramienta será implementada usando Java Servlets.	-	NF
9	La herramienta será implementada en Java SE 7.0.	-	NF
10	El servidor de aplicaciones será Tomcat 8.0	-	NF

Fuente: propia

1.2 Diseño

Para especificar el diseño de la herramienta, se utilizará el modelo 4+1.

Vista de Casos de Uso

Para esta herramienta sencilla se ha identificado un caso de uso: el de simplificación de textos, tal como se puede apreciar en la figura 5.1.

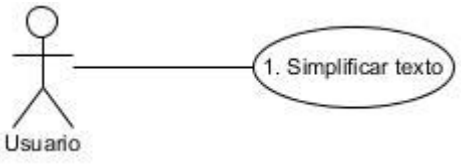

Fig. 5.1. Casos de uso de la herramienta de simplificación de textos.
Fuente: propia.

Este caso de uso está especificado con mayor detalle en la tabla 5.2.

Tabla 5.2. Especificación del caso de uso 1.

Código	1
Caso de uso	Simplificar texto.
Actor	Usuario.
Descripción	Se ingresa un texto y se devuelve el texto simplificado.
Precondición	Servidor de aplicaciones Tomcat encendido. Texto original carece de errores ortográficos y gramaticales.
Flujo básico	1. El usuario ingresa a su navegador web. 2. El usuario ingresa la dirección de la aplicación web. 3. La aplicación presenta al usuario un formulario con dos cuadros de texto: uno para el texto original y otro para el texto simplificado. 4. El usuario copia el texto original en el cuadro que se indica para tal fin. 5. El usuario da clic sobre el botón “Simplificar”. 6. La aplicación ejecuta las operaciones de simplificación sobre el texto. 7. La aplicación muestra al usuario el texto simplificado.
Flujo alternativo	No tiene.
Postcondición	Se tiene un texto simplificado.

Fuente: propia

Vista de desarrollo

Esta vista puede ser resumida en el diagrama de componentes de la figura 5.2. En esta figura puede observarse dos componentes claramente definidos: el "*frontend*" y el "*backend*". En el primero, se encuentran los elementos visuales de la aplicación. En el segundo, se encuentra la lógica necesaria para implementar las operaciones de simplificación y los clasificadores. Asimismo, puede observarse que la comunicación entre ambos componentes se da a través de servicios web.

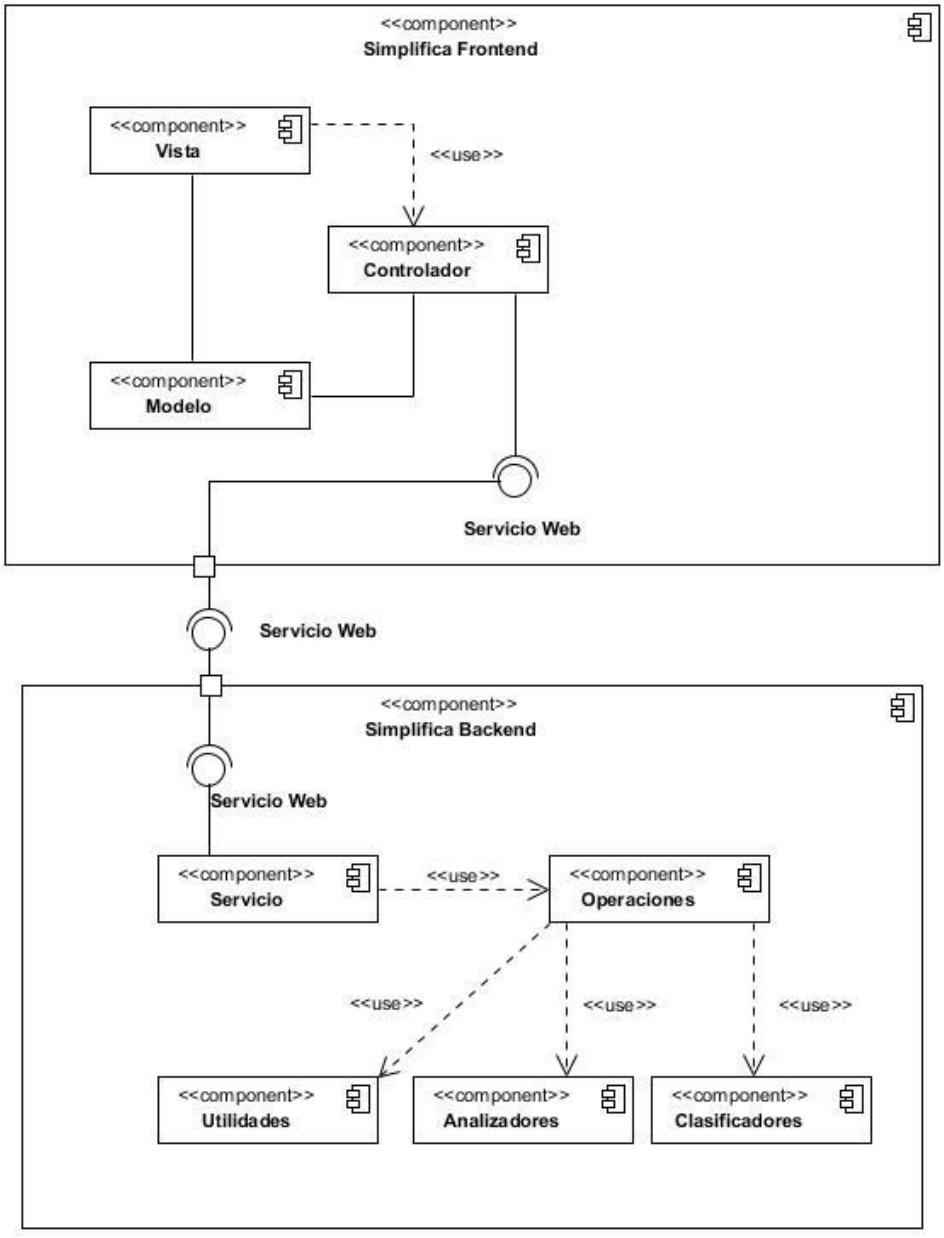

Fig. 5.2. Diagrama de componentes de la aplicación.

Fuente: propia.

Vista Lógica

En la figura 5.3 se muestra el diagrama de clases que se utilizó para implementar la herramienta. Puede observarse que se utilizó el patrón de fábrica abstracta para las clases correspondientes a los clasificadores y a las operaciones. Asimismo, se utilizó un patrón "*Singleton*" para la clase Módulos, que contiene los módulos de análisis de Freeling.

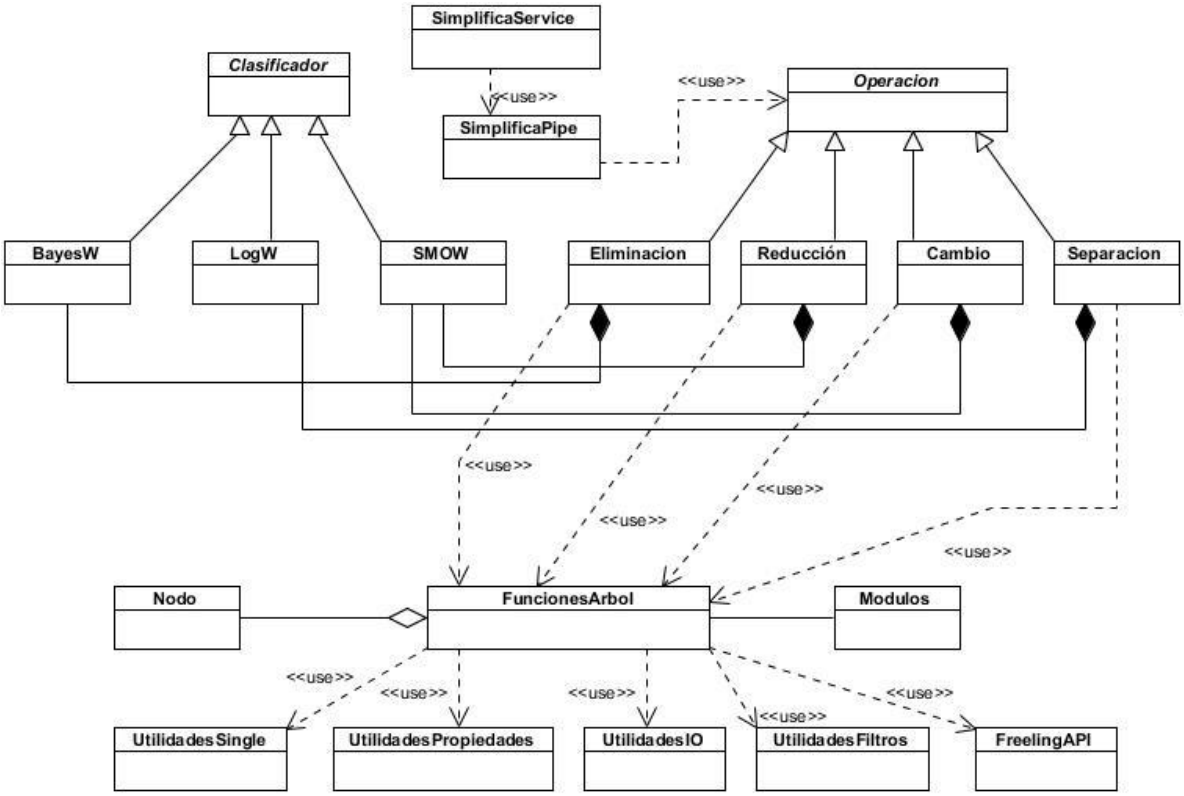

Fig. 5.3. Diagrama de clases del "backend" de la aplicación.
Fuente: propia.

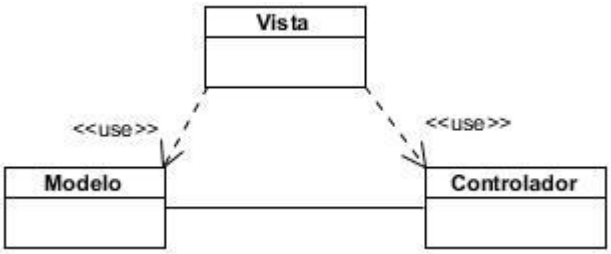

Fig. 5.4. Diagrama de clases del "frontend" de la aplicación.
Fuente: propia.

Asimismo, en la figura 5.4 se muestra el diagrama de clases del “frontend” de la aplicación. No hay mucho que mencionar al respecto, pues es un simple diagrama de un modelo MVC.

Vista de procesos

La lógica seguida para aplicar las operaciones de simplificación se ve reflejada en el diagrama de actividades de la figura 5.5.E En ella, puede apreciarse que las operaciones se aplican secuencialmente sobre el texto y que sólo son llevadas a cabo si así lo decide el clasificador. En este diagrama, los nodos de decisión representan a los clasificadores, que como ya se indicó, son quienes toman la decisión de aplicar o no la operación respectiva.

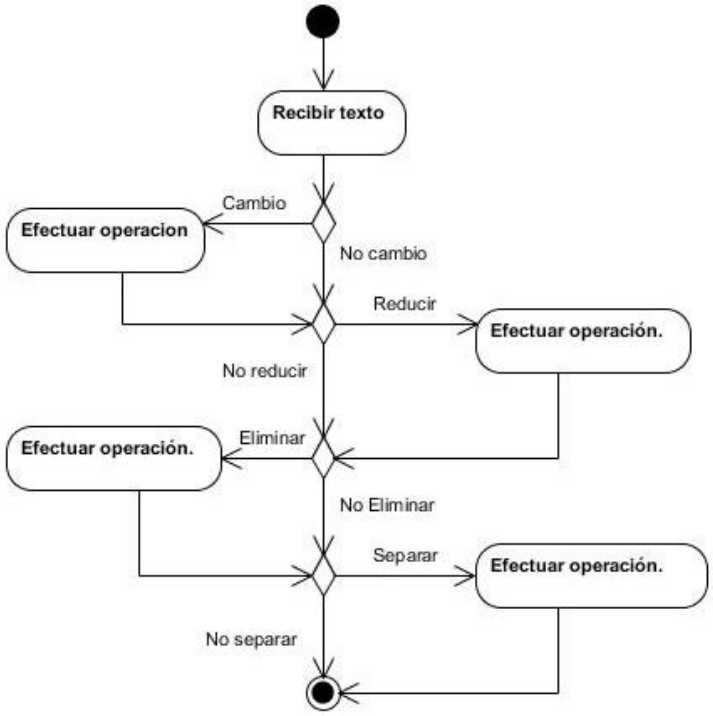

Fig. 5.5. Diagrama de actividades para la simplificación de textos.
Fuente: propia.

Vista de despliegue

En la figura 5.6 puede apreciarse el diagrama de despliegue de la solución. Al tratarse de una aplicación web (cliente/servidor), se reconocen dos nodos principales: el servidor, que contiene tanto el “frontend” como el “backend” comunicados por servicios web; y el cliente, que es una PC o laptop que tenga un explorador de internet que le permita acceder a la aplicación vía TCP/IP.

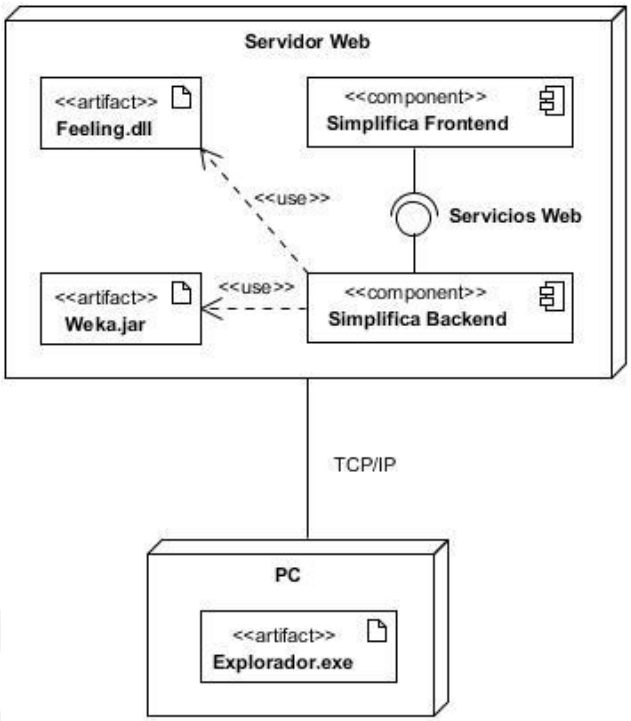

Fig. 5.6. Diagrama de despliegue de la aplicación.
Fuente: propia.

1.3 Implementación

1.3.1 Vista ("Frontend")

Tal como se mencionó repetidas veces, esta parte de la herramienta fue desarrollada siguiente una arquitectura MVC. No se destinó muchos recursos a esta parte de la implementación debido a que no era parte trascendental del proyecto. No obstante, es rescatable mencionar que se trabajó con tecnología de servlets de Java combinada con HTML y CSS para dar una vista agradable al usuario. En la figura 5.7 puede apreciarse un prototipo de la pantalla.

En este prototipo puede reconocerse dos elementos principales: un área de texto para que el usuario ingrese su texto original y otra área de texto para que el usuario visualice el texto simplificado. Asimismo, en la parte inferior hay un botón ("Simplificar texto") para iniciar el proceso.

Fig. 5.7. Captura de pantalla de la vista web de la aplicación.
Fuente: propia.

1.3.2 Operaciones de simplificación ("Backend")

El principal objetivo en esta parte fue la implementación de las operaciones de simplificación. En vista de la gran diversidad de estructuras gramaticales, se trabajó con el árbol de dependencia (descrito en el marco conceptual) que genera Freeling cuando analiza una oración. Por conveniencia, dicha estructura fue trasladada a una representación propia de dicho árbol en la que cada nodo almacenaba la información sintáctica de cada palabra de la oración y los hijos del mismo.

Las operaciones implementadas fueron: separación de oraciones compuestas coordinadas (coordinadas copulativas, coordinadas adversativas exclusivas, coordinadas distributivas, coordinadas explicativas, coordinadas consecutivas), separación de oraciones compuestas subordinadas adjetivas explicativas (también conocidas como cláusulas relativas), separación de construcciones de participio, separación de construcciones con elipsis en el objeto directo e indirecto; cambio de la estructura de reporte; eliminación de oraciones y reducción de oraciones (eliminación de elementos internos de la oración). Para mayor ejemplificación, ver el **anexo 10.**

Asimismo, otra parte importante fue la incorporación de los clasificadores para que tomen la decisión de efectuar una operación o no. Esto se llevó a cabo usando el patrón de fábrica abstracta: se implementó una fábrica de clasificadores y otra de

operaciones. Por tal motivo, se definió una clase abstracta para un clasificador y otra clase abstracta para una operación.

Los clasificadores fueron concebidos para actuar como envoltorios ("wrappers") de las clases expuestas por el API de Weka. Gracias a esto, fue posible personalizar los métodos para entrenar y aplicar los clasificadores de acuerdo a las necesidades del proyecto. Cada clasificador fue una implementación de la clase abstracta mencionada en el párrafo anterior. La especificación de dicha clase abstracta es la siguiente:

```
public abstract class Clasificador {

    public abstract void entrenar (conjuntoEntrenamiento, opciones) ;

    public abstract Instances aplicar (conjuntoAplicar);

}
```

Por otro lado, las clases correspondientes a las operaciones fueron implementadas de manera tal que soportaran el uso de cualquier algoritmo de clasificación para decidir si dicha operación debía aplicarse o no. Asimismo, fueron implementadas para aplicar la operación correspondiente. Nuevamente, cada operación fue una implementación de una clase abstracta cuya especificación es la que sigue:

```
public abstract class Operación {

    protected Clasificador clasificador;

    public abstract Instances decideOperacion (conjuntoEtiquetar);

    public abstract String aplicaOperacion (texto);

}
```

1.4 Evaluación de resultados

Para evaluar los resultados obtenidos, se tomó una muestra de 50 textos del corpus (2 grupos de 25 textos cada uno) y sus versiones simplificadas manualmente y se calculó el índice de perspicuidad sobre cada uno usando la herramienta INFLESZ. Se calculó:

$$Dif = Persp(Simp.Manual) - Persp(Original) \dots (\boldsymbol{Ec.5.1})$$

Y con esta diferencia se formó la línea base contra la cual se comparó la misma diferencia para cada texto simplificado con la herramienta:

$$Dif = Persp(Simp.Auto) - Persp(Original) \quad \ldots(\boldsymbol{Ec.5.2})$$

Los resultados obtenidos, que fueron tabulados y graficados, se pueden apreciar en las figuras 5.8 y 5.9.

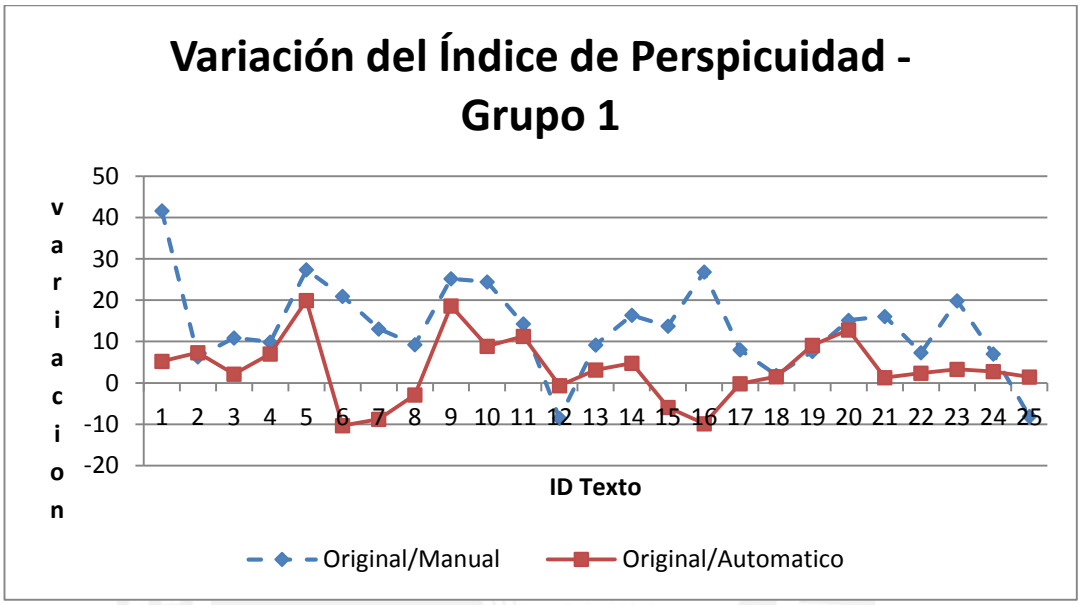

Fig. 5.8. Variación del índice de perspicuidad para las primeras 25 muestras.
Fuente: propia.

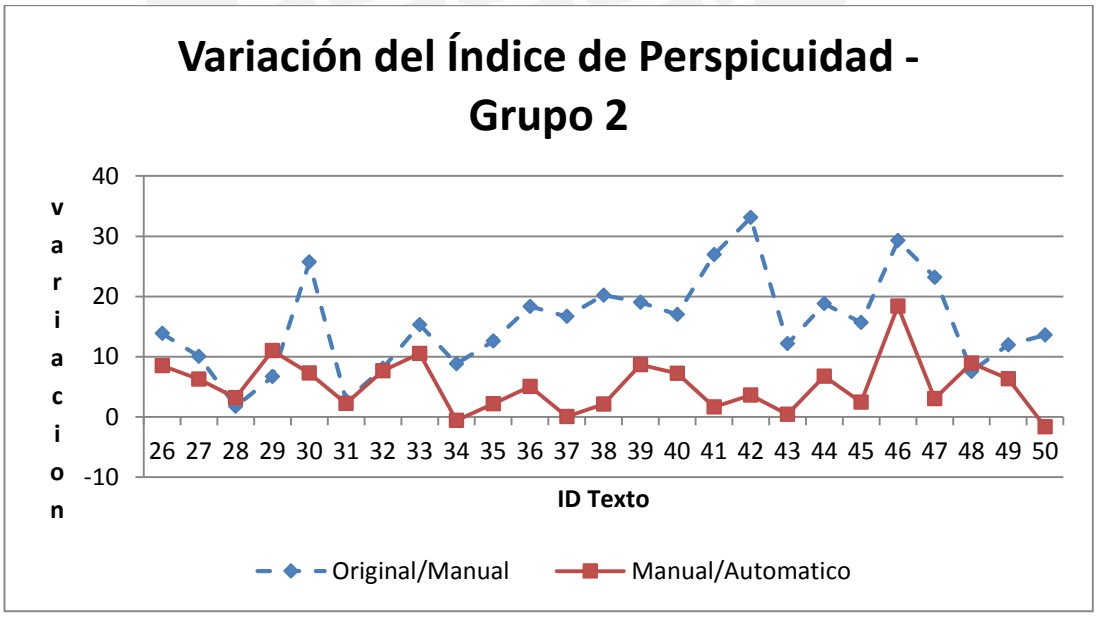

Fig. 5.9. Variación del índice de perspicuidad para las últimas 25 muestras.
Fuente: propia.

Como puede apreciarse en las figuras, el sistema contribuyó a reducir la complejidad de los textos en la mayoría de casos (variación > 0). No obstante, se observó también que la simplificación manual (líneas punteadas) tuvo un mejor efecto que la simplificación automática (línea sólida). Para la simplificación automática, el índice aumentó en un promedio de 6.25 unidades con respecto al original; mientras que en los casos de simplificación manual, éste aumentó en un promedio de 15.45 unidades.

Esta diferencia era de esperarse por varios motivos. En primer lugar, el proceso automático carece de un elemento que sí tiene el proceso manual: un agente humano con conocimiento y experiencia. Éste puede, en base a su experiencia, llevar a cabo operaciones de simplificación que una máquina no es capaz de llevar a cabo. En segundo lugar, el sistema sólo lleva a cabo una simplificación a nivel sintáctico; es decir, no incluye las operaciones de sustitución léxica que son mucho más frecuentes y que tienen una contribución importante a la reducción de complejidad de un texto.

1.5 Conclusiones del capitulo

En este capítulo se describió el análisis, diseño, implementación y evaluación de una herramienta para simplificar textos en español. En esta aplicación se reconocieron dos grandes componentes: el visual (frontend) y el lógico (backend). El primero fue construido usando tecnología de servlets con una arquitectura MVC. El segundo fue implementado usando el patrón de fábrica abstracta y su principal objetivo fue integrar los clasificadores al sistema e implementar las operaciones de simplificación. La integración entre ambos componentes se dio a través de servicios web.

El desempeño de la aplicación fue evaluado comparando diferencias del índice de Flesch-Szigriszt (índice de perspicuidad) entre los textos simplificados y los textos originales. Se pudo observar que en efecto, el sistema contribuyó a mejor dicho índice para la mayoría de textos, pero que la simplificación manual fue más efectiva. Esta diferencia se atribuyó a dos hechos: la experiencia del agente humano y la ausencia de un módulo de simplificación sintáctica.

A pesar de tener un desempeño inferior, el sistema cumple con su objetivo de reducir la complejidad de los textos que recibe como entradas al aplicar operaciones de simplificación. Si bien en la mayoría de casos se tienen textos

gramaticalmente correctos, en algunos casos dicha característica se pierde, principalmente cuando el analizador no genera el árbol de dependencia correcto.

CAPITULO 6

1 Conclusiones

El acceso a la información es un derecho reconocido por la ONU que tiene todo ser humano. Dicho acceso implica que todas las personas deben de ser capaces de entender la información que está disponible en los distintos medios de información escrita. No obstante, existen varios grupos de personas que no pueden entender dicha información debido a que, por diversos motivos, tienen un nivel bajo de comprensión lectora. Estas personas no pueden entender estos textos debido a que son muy complejos para ellos.

A pesar de que en teoría la información escrita debería ser comprensible para todos, esto no ocurre debido a que la simplificación manual es muy costosa en términos de tiempo y dinero. Por ello, la automatización de dicho proceso es una tarea de interés dentro del campo del Procesamiento de Lenguaje Natural (PLN). En tal sentido, se han reportado investigaciones sobre la automatización de simplificación de textos en idiomas como inglés, portugués, italiano, etc. No obstante, la investigación para el español es bastante reducida y solo se encontró el simplificador del proyecto Simplext. Esta investigación desarrolló un simplificador basado principalmente en reglas; aunque al final del mismo llegaron a incorporar un filtro estadístico sólo para la separación de cláusulas relativas.

Es por ello que en este proyecto se propuso la implementación de un módulo de simplificación sintáctica que incorpore elementos de aprendizaje supervisado de máquina que permita simplificar textos en español. En vista que no existe un corpus de textos en español lo suficientemente grande como para crear un agente que aprenda cómo llevar a cabo cada operación de simplificación, se adoptó el enfoque del proyecto PorSimples (simplificación de textos en portugués) de entrenar clasificadores binarios que tomen la decisión de efectuar o no una operación en particular.

Para llevar a cabo el proyecto, se requería de un corpus paralelo formado por textos en español alineado con sus versiones simplificadas manualmente. Por ello, se contactó con el grupo de investigación del proyecto Simplext, quienes dieron su autorización para el uso del corpus que recolectaron durante su proyecto. Este corpus fue formado por 190 noticias de la agencia Servimedia; simplificado manualmente por editores especializados del grupo DILES de la Universidad

Autónoma de Madrid; y finalmente alineado automáticamente por un algoritmo no supervisado de aprendizaje de máquina (Saggion y Bott, 2011b).

Este corpus fue enriquecido con la información de las operaciones de simplificación que se llevaron a cabo oración por oración. Para facilitar esta tarea inherentemente manual, se implementó una herramienta de apoyo al anotador, la cual resultó de gran utilidad pues permitió disminuir considerablemente el tiempo estimado. Del análisis de frecuencias de las operaciones anotadas, se concluyó que solo eran representativas las operaciones de separación, eliminación, reducción y cambio, que juntas con la opción de mantener la oración tal cual, explican el 97% de los cambios en los textos del corpus.

Adicionalmente, se identificó un conjunto de 37 características que podían ser extraídas de los textos para formar un vector de atributos. Este vector es de crucial importancia porque es el dato de entrada para el clasificador. Este conjunto estuvo formado por características utilizadas por el proyecto Simplext, el proyecto PorSimples y un conjunto adicional de características identificadas durante el análisis preliminar de los textos. Los extractores fueron implementados en Java y probados con JUnit.

Posteriormente, con el apoyo de la herramienta Weka, se llevó una experimentación numérica que permitió definir el clasificador más adecuado para cada operación y el vector de características más adecuado para cada clasificador. Para ello, en la experimentación, se comparó las medidas F promedio utilizando la prueba T para pares ("*paired-T test*"). Se concluyó que para la operación de eliminación debía usarse el clasificador simple de Bayes con el conjunto reducido de atributos; para la operación de separación, un clasificador logístico con el conjunto reducido de atributos; y, para las operaciones de cambio y reducción, una máquina de soporte vectorial (SMO) con el conjunto completo de características.

Finalmente, se implementó una aplicación web que simplifica textos que recibe del usuario como dato de entrada. Para simplificar los textos, se utilizaron los clasificadores anteriormente señalados para determinar si efectuar o no una operación y, en caso la respuesta era positiva, aplicar la operación sobre la oración. Para implementar dichas funciones, se analizó y modificó el árbol de dependencia generado por cada oración. Luego de analizar las variaciones del índice de perspicuidad, se pudo concluir que la herramienta sí contribuye a la reducción de la complejidad de los textos, aunque con un menor grado de eficacia que un editor humano. Esta diferencia era la esperada debido a que no se incluyó un módulo de

simplificación léxica y porque un agente humano tiene experiencia y conocimiento que una máquina carece.

Es por ello que en términos generales, se puede afirmar que se cumplió con el objetivo general de este proyecto: se implementó un módulo de simplificación a nivel sintáctico que lleva a cabo operaciones de simplificación cuando un clasificador binario entrenado específicamente para dicha operación indica que ésta debe efectuarse. Este módulo permitió reducir la complejidad de los textos, medida con el índice de perspicuidad.

2 Recomendaciones y trabajo futuro

Este trabajo, como todo esfuerzo humano, es sujeto a perfeccionamiento. En tal sentido, hay aspectos en la presente investigación que pueden ser mejorados. Estas mejoras permitirán disminuir la brecha existente entre el desempeño del sistema comparado con el de un editor humano. En primer lugar, sería recomendable integrar un módulo de simplificación léxica, que es una operación mucho más sencilla y a su vez más frecuente que la simplificación sintáctica

Por otro lado, como ya se mencionó, la anotación del corpus con las operaciones de simplificación fue un proceso manual; por ello, susceptible a errores. Por falta de recursos, fue llevada a cabo solo una vez y por una sola persona, por lo que existe la posibilidad que existan algunos errores en la anotación. Para minimizar este riesgo, sería recomendable validar el resultado con un conjunto de editores expertos; o en su defecto, solicitarles a ellos que realicen la anotación.

Asimismo, sería recomendable incrementar el tamaño del corpus por varios motivos. En primer lugar, porque permitiría aplicar un enfoque basado 100% en reglas; es decir, que permita al sistema inducir las reglas para simplificar. Incluso si no se deseara optar por la opción anterior, también permitiría identificar un mayor número de características extraíbles para formar el vector de atributos.

Para la selección de clasificadores, en la presente investigación, se trabajó con los parámetros predeterminados ofrecidos por Weka; no obstante es posible que el desempeño de los clasificadores mejore significativamente con otras configuraciones. Por ello, en un futuro, se trabajará en la calibración de parámetros de los algoritmos de clasificación. Los resultados de tal investigación tendrán el potencial de cambiar los resultados obtenidos en este documento.

Asimismo, en un futuro inmediato, se trabajará en la aplicación de filtros adicionales para mejorar la gramaticalidad de las oraciones obtenidas. Esto es necesario porque en algunas ocasiones, el resultado devuelto por la herramienta no es correcto gramaticalmente. Este error es debido principalmente a la generación de un árbol de dependencia errado, lo cual puede evitarse aplicando más filtros.

Finalmente, en un trabajo futuro se incorporará métricas de evaluación de complejidad en etapas intermedias del proceso con la finalidad de simplificar el texto hasta que alcance un nivel de dificultad deseado, a diferencia del sistema actual, que simplifica el texto al máximo.

REFERENCIAS BIBLIOGRÁFICAS

ALLINGTON, Richard.

2002 "You Can't Learn Much from Books You Can't Read". Educational Leadership, 2002, V.60, No.3, p.16.

ALPAYDIN, Ethem.

2010 Introduction to Machine Learning. Cambridge: The MIT Press. Pp.1-19.

ALUÍSIO, Sandra María; SPECIA, Lucia; PARDO, Thiago, MAZIERO, Erick; FORTES, Renata.

2008 "Towards Brazilian Portuguese Automatic Text Simplification Systems". Proceedings of the Eight ACM Symposium on Document Engineering. San Pablo, 2008, pp. 240-248.

ALUÍSIO, Sandra María; GASPERIN, Caroline.

2010 "Fostering Digital Inclusion and Accesibility: The PorSimples project for Simplification of Portuguese Texts". Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas. Los Ángeles, 2010, pp. 46-53.

ANULA, Alberto.

2008 "Lecturas adaptadas a la enseñanza del español como L2: variables lingüísticas para la determinación del nivel de legibilidad". La evaluación en el aprendizaje y la enseñanza del español como LE/L2. Alicante, 2008, pp. 162-170.

BARRIO-CANTALEJO, I. M., SIMÓN-LORDA, P., MELGUIZO, M., ESCALONA, I., MARIJUÁN, M., & HERNANDO, P.

2008. "Validación de la Escala INFLESZ para evaluar la legibilidad de los textos dirigidos a pacientes" An. Sist. Sanit. Navar. 31(2), pp.135-152.

BIRD, Steven; KLEIN, Ewan; LOPER, Edward.

2009 Natural Language Processing with Python. s/c: Natural Language Toolkit.

CANDIDO JUNIOR, Arnaldo; MAZIERO, Erick; GASPERIN, Caroline; PARDO, Thiago; SPECIA, Lucia; ALUISIO, Sandra.

2009 "Supporting the Adaptation of Texts for Poor Literacy Readers: a Text Simplification Editor for Brazilian Portuguese". Proceedings of the NAACL HLT Workshop on Innovative use of NLP for building Educational Applications . Colorado, 2009, pp. 34-42.

CANDIDO JUNIOR, Arnaldo; COPESTAKE, Ann.

2011 "Towards an on-demand Simple Portuguese Wikipedia". Proceedings of the 2nd Workshop on Speech and Language Processing for Assistive Technologies. Edinburgh, 2011, pp. 137-147.

CANDIDO JUNIOR, Arnaldo.

2013 Análise bidirectional da língua na simplificaçao sintática em textos do portugués voltada à acessibilidade digital. Tesis de Doctorado. San Carlos: Instituto de Ciencias Matemáticas y de Computación de la Universidad de Säo Paulo (ICMC-USP).

COMMON CORE STATE STANDARDS (CCSS)

s/f Common Core State Standards for English Language Arts & Literacy in History/Social Studies, Science, and Technical Subjects. Appendix A: Research Supporting Key Elements of the Standards. Consulta: 09-04-2014.

En línea: http://www.corestandards.org/assets/Appendix_A.pdf.

CUNNINGHAM, Pádraig; CORD, Matthieu; DELANY, Sarah Jane.

2008 Machine Learning Techniques for Multimedia. s/c: Springer. Pp. 21-22.

DALE, R., MOISL, H., & SOMERS, H.

2000 *Handbook of Natural Language Processing.* Nueva York: Marcel Dekker Inc. Pp. 403-404.

DRNDAREVIĆ, Biljana; SAGGION, Horacio.

2012 "Towards Automatic Lexical Simplification in Spanish: An Empirical Study". Workshop Predicting and Improving text readability for target reader populations. Montréal, 2012, pp. 8-16.

FEBLOWITZ, Dan; KAUCHAK, David.

2013 "Sentence Simplification as Tree Transduction". Proceedings of the 2nd Workshop on Predicting and Improving Text Readability for Target Reader Populations. Sofía, 2013, pp. 1-10.

FISHER, Douglas; FREY, Nancy.

2012 "Text Complexity and Close Readings". Engaging the adolescent learner. Newark, 2012, pp. 1-15.

FREELING.

2012 FreeLing Features. Consultado: 28-05-2014.

En línea: http://nlp.lsi.upc.edu/freeling/index.php?option=com_content&task=view&id=12&Itemid=41

GARCÍA MÁRQUEZ, Gabriel.

1967 Cien años de soledad. Bogotá: Grupo Editorial Norma.

JAIN, A.K.; MURTY, M.N.; FLYNN, P.J.

1999 "Data Clustering: A Review". ACM Computing Surveys, V.31, No.3, 1989. Pp. 264-323.

GASPERIN, C., SPECIA, L., PEREIRA, T. F., & ALUISIO, S. M.

2009. "Learning when to Simplify Sentences for Natural Text Simplification". *Proceedings of ENIA*, pp. 809-818

JURAFSKY, Daniel; MARTIN, James H.

2007 Speech and Language Processing: An introduction to Natural Language Processing, Computations Linguistics, and Speech Recognition. Upper Saddle River: Pearson Prentice Hall.

LEGIBILIDAD.

2007 "¿Qué es la legibilidad?" Consulta: 08-04-2014.

En línea: http://legibilidad.com/home/acercade.html#legibilidad.

LIDDY, E.D.

2001 Natural Language Processing. Encyclopedia of Library and Information Science. Segunda Edición. Nueva York: Marcel Decker Inc.

MANNING, Christopher; SHUTZE, Hinrich.

1999 Foundations of Statistical Language Processing. Segunda edición. Londres: Instituto Tecnológico de Massachusetts.

MINISTERIO DE EDUCACIÓN, GOBIERNO DE PERÚ (MINEDU).

2013 Evaluación Censal de Estudiantes 2013. Consultado: 04-04-2013.

En línea: http://umc.minedu.gob.pe/?p=1766.

MITKOV, R.

2009. "The Oxford Handbook of Computational Linguistics". Nueva York: Oxford University Press. Pp.219 – 220.

NATIONAL APHASIA ASSOCIATION.

2011 "Aphasia Definitions". Consultado: 10-04-2014.

En línea: https://www.aphasia.org/content/aphasia-definitions

NATIONAL CENTER FOR LEARNING DISABILITIES (NCLD).

2014 "General LD Info". Consultado: 15-05-2014.

En línea: http://ncld.org/types-learning-disabilities/what-is-ld

ORGANIZATION FOR ECONOMIC CO-OPERATION AND DEVELOPMENT (OECD).

2012 Pisa 2012 Results: What students know and can do. "Student Performance in Mathematics, Reading and Science", Vol. I.

PROJECT MANAGEMENTE INSTITUTE (PMI).

2013 A Guide to the Project Management Body of Knowledge (PMBOK), Fifth Edition. Pennsylvania: Project Management Institute.

PROVOST, Foster; FAWCETT, Tom

2013 Data Science for Business: what you need to know about data mining and data-analytic thinking. Estados Unidos: O'Reilly. 386 pp.

QUISPESARAVIA, André; PEREZ, Walter.

2013 Herramienta de Clasificación automática de Complejidad de Textos en Español. Tesis de licenciatura. Lima: Pontificia Universidad Católica del Perú, Ingeniería Informática.

REAL ACADEMIA ESPAÑOLA.

2001 "Lenguaje". Diccionario de la lengua Española. Decimosegunda edición. Consulta: 08-04-2014.

En línea: http://lema.rae.es/drae.

REAL ACADEMIA ESPAÑOLA.

2001 "Lengua". Diccionario de la lengua Española. Decimosegunda edición. Consulta: 08-04-2014.

En línea: http://lema.rae.es/drae.

REAL ACADEMIA ESPAÑOLA.

2001 "Complejidad". Diccionario de la lengua Española. Decimosegunda edición. Consulta: 08-04-2014.

En línea: http://lema.rae.es/drae.

REAL ACADEMIA ESPAÑOLA.

2001 "Complejo". Diccionario de la lengua Española. Decimosegunda edición. Consulta: 08-04-2014.

En línea: http://lema.rae.es/drae.

SAGGION, Horacio.

2008 "Automatic Summarization: An Overview". H. Revue Francaise de Linguistique Apliquee, 2008, V.13, No.1, pp. 63-81.

SAGGION, Horacio; BOTT, Stefan.

2011a “Spanish Text Simplification: An Exploratory Study”. H.Journal of the Spanish Society for Natural Language Processing. Barcelona, 2011, V. 47, No. 2, pp. 87-95.

SAGGION, Horacio; BOTT, Stefan.

2011b “An unsupervised alignment algorithm for text simplification corpus construction”. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, 2011, pp. 20-26.

SAGGION, Horacio; GÓMEZ-MARTÍNEZ, Elena; ETAYO, Esteban; ANULA, Alberto; BOURG, Lorena.

2011 “Text Simplification in Simplext: Making Texts more Accessible”. Procesamiento del Lenguaje Natural, 2011, No.47, pp. 341-342.

SAGGION, Horacio; BOTT, Stefan; FIGUEROA, David.

2012 “A Hybrid System for Spanish Text Simplification”. Workshop on Speech and Language Processing for Assistive Technologies. Montreal, 2012, pp.75-84.

SAGGION, Horacio; DRNDAREVIC, Biljana; STAJNER, Sanja; BAUTISTA, Susana.

2013 “Automatic Text Simplification in Spanish: A Comparative Ecaluation of Complementing Modules”. Proceedings of the 12th International Conference on Intelligent Text Processing and Computational Linguistics. Lecture Notes in Computer Science, Samos, 2013, pp. 488-500.

SAGGION, Horacio; BOTT, Stefan.

2014 “Text simplification resources for Spanish”. Lang. Resources & Evaluation, 2014, V.48, No.93, pp.93-120.

SAGGION, Horacio; FUNK, Adam.

2009 “Extracting Opinions and Facts for Busuiness Intelligence”. RNTI Journal, V.17, pp. 119-146.

STAJNET, Sanja; SAGGION, Horacion, Horacio.

2013 “Adapting text simplification decisions to different text genres and target users”. Procesamiento del Lenguaje Natural, Nro. 51, pp. 135-142.

UNITED NATIONS (UN).

1949. The Universal Declaration of Human Rights. (Consultado: 23-03-2014).

En línea: http://www.un.org/en/documents/udhr

VARGAS, Irvin.

2013 Herramienta informática de apoyo a la escritura de resúmenes de textos científicos en español. Tesis de Licenciatura. Lima: Pontificia Universidad Católica del Perú, Ingeniería Informática.

VELLUTINO, Frank R.; FLETCHER, Jack M.; SNOWLING, Margaret; SCANLON, Donna.

2004 "Specific reading disability (dyslexia): what have we learned in the past four decades?" Journal of Child Psychology and Psychiatry. 2004, V. 45, No. 1, pp. 2-40.

WEKA.

2013 Weka 3: Data Mining Software in Java. Consultado: 28-05-2014.

En línea: http://www.cs.waikato.ac.nz/ml/weka/index.html.

WEB ACCESSIBILITY INITIATIVE (W3C)

2012 Introduction to Web Accessibility. Consultado el 18-04-2014.

En línea: http://www.w3.org/WAI/intro/accessibility.php

WOODSEND, Kristian; LAPATA, Mirella.

2011a "WikiSimple: Automatic Simplification of Wikipedia Articles". Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence. San Francisco, 2011, pp. 927-932.

WOODSEND, Kristian; LAPATA, Mirella.

2011b. "Learning to Simplify Sentences with Quari-Synchronous Grammar and Integer Programming". Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. Edinburgh, 2011, pp. 409-420.

YATSKAR, Mark; BO, Pang; DANESCU-NICULESCU-MIZIL, Cristian; LEE, Lillian.

2010 "For the sake of text simplicity: Unsupervised extraction of lexical simplifications from Wikipedia". Proceedings of the Annual Meeting of the North American Chapter of the Association for Computational Linguistics, Los Angeles, 2010, pp. 365-368.

ZHU, Zhemin, BERNHARD, Delphine; GUREVYCH, Iryna.

2010 "A monolingual tree-based transformation model for sentence simplification" Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, 2010, pp. 1353-1361.