CAMPUS
DE EXCELENCIA
INTERNACIONAL

Graduado en Ingenería Informática

Universidad Politécnica de Madrid

Escuela Técnica Superior de
Ingenieros Informáticos

TRABAJO FIN DE GRADO

Desambiguación de acrónimos biomédicos en español mediante técnicas de Machine Learning

Autor: Ignacio Rubio López

Director: Ernestina Menasalvas Ruiz

MADRID, JUNIO 2016

Resumen

En la última década debido a la gran cantidad de información generada con medios tecnológicos, se ha determinado la información no estructurada como un gran nicho de conocimiento implícito. Nuevas técnicas de la tecnologías de la información tienen como objetivo extraer conocimiento explícito de información no estructurada, como por ejemplo las técnicas de Procesamiento de Lenguaje Natural (PLN) más conocida como Natural Language Processing (NLP). Estas técnicas, ayudadas por otras técnicas de Machine Learning (Aprendizaje automático) son capaces de realizar tareas de predicción y clasificación sobre elementos de los textos con bastante precisión. En este trabajo se desarrollará como se han utilizado las técnicas antes mencionadas para la tarea de desambiguación de acrónimos.

Con la digitalización de los documentos en el ámbito sanitario, la extracción de información de las notas clínicas puede ser extraída y utilizada en infinidad de aplicaciones. Por desgracia, para realizar una extracción de esta información de forma satisfactoria se requiere la resolución de diversos problemas que presenta la información no estructurada. La ambigüedad es un problema crucial y en concreto en este trabajo se resuelve la ambigüedad introducida por los acrónimos y siglas en notas clínicas en español.A pesar de haber casi 500 millones de hispano-hablantes, apenas se han desarrollado algoritmos de este tipo, por lo tanto este trabajo aborda una problemática poco desarrollada.

En este trabajo se ha planteado el problema de desambiguación como un problema de clasificación, es decir, se van a extraer diversas características lingüísticas, como por ejemplo los conceptos biomédicos que rodean al acrónimo, o el análisis morfológico de las palabras que le rodea. El algoritmo va a tratar de determinar si con esas características una posible definición del acrónimo es correcta o no. Por lo tanto las tareas principales que va a desarrollar este trabajo son la selección y extracción de características, así como la selección de la técnica de Machine Learning más adecuada para la tarea de desambiguación.

Los recursos utilizados para la realización de este trabajo constan de 150 notas clínicas en español, procedentes de diversos hospitales que generan más de 30.000 datos para analizar. Además se han utilizado herramientas como UIMA para la generación de metadatos en las notas clínicas junto con UMLS para añadir más información sobre los conceptos médicos. Para las técnicas de Machine Learning se ha utilizado la herramienta Weka que contiene múltiples algoritmos de Machine Learning y algoritmos de validación.

Abstract

During the last ten years, digitalized information has grown exponentially, and, in order to extract implicit information from non-structured data, new technologies have been developed. Natural Language Processing (NLP) techniques are used to analyze digital texts and extract multiple types of information, which may be useful for extracting implicit information. In the same vein, Machine Learning techniques are regularly used in NLP to solve different issues during text analysis processes. In this project, the use of Machine Learning techniques to solve the acronym disambiguation task will be discussed.

Although there are nearly 500 million Spanish speakers worldwide, there seems to be no algorithm for biomedical acronym disambiguation in medical texts written in that language. The overuse of acronyms in clinical notes makes the NLP task extremely difficult, due to the fact that acronyms introduce an enormous ambiguity. The approach discussed in this project solves the acronym's ambiguity issue successfully by using contextual elements around the acronyms such as Part of Speech and surrounding biomedical concepts.

The disambiguation problem has been defined as a classification task. The algorithm will try to classify whether an acronym is standing for a definition or not. Different features will be selected and extracted from the texts. Furthermore, different Machine Learning algorithms will be selected and validated to find the most suitable algorithm for the disambiguation task.

In this project, 150 clinical notes in Spanish from different hospitals have been used. These notes have generated more than 30,000 entries to be analyzed. Additionally, the tools used in this project have been: UIMA, to generate metadata from the clinical notes; UMLS, to provide the information related to the biomedical field, and Weka, to apply Machine Learning algorithms, create models and validate the models created.

Índice

Índice de figuras

Índice de tablas

1 INTRODUCCIÓN

1.1 Introducción

Las tecnologías de la información han supuesto cambios profundos en la manera de hacer nuestras tareas, comportarnos y comunicarnos. Estas tecnologías tienen una característica especial: nuestros trabajos, conexiones y comunicaciones generan datos, datos que se almacenan y contienen una gran cantidad de información relacionada con nuestros gustos, costumbres y relaciones personales. Esto hace que la cantidad de información digital crezca sin límites ya que cada vez más y más gente está conectada, en 2014 cerca de la mitad de la población mundial ya estaba conectada a internet según el estudio de [1] Internet World Stats. Para hacerse un ligera idea, la cantidad aproximada de información que se manejaba en Internet en 2013 era aproximadamente 1.2 zettabytes (1021 bytes) y la cantidad de información que se genera por segundo en Internet de unos 22000 gigabytes. Wal-Mart, una multinacional de tiendas americana maneja más de un millón de transacciones de clientes cada hora, alimentando bases de datos de más de 2,5 petabytes. Facebook maneja ya 40.000 millones de fotos [2].

Toda esta información se convierte en un nicho de interés en muchos ámbitos desde el comercial hasta el académico. El uso de esta información nos puede llegar a permitir la toma de decisiones, predicción de sucesos, así como el descubrimiento de conocimiento implícito. Esta desorbitada cantidad de datos no puede ser analizada con medios tradicionales, es aquí donde la propia tecnología toma un papel fundamental no solo para generar esos datos sino también para analizarlos. Con este caldo de cultivo, dos disciplinas se unen, la ya madura estadística con la reciente informática, surgiendo así el concepto "Data Science" que se encarga de dar sentido a esa gran cantidad de información desestructurada. Sin embargo esta ciencia es relativamente joven, comenzando como una ciencia que se dedicaba únicamente al manejo de los datos mientras que la representación y la relación de esta se delegaba a otras ciencias, acabando como una ciencia capaz de encontrar nuevas fuentes de datos, valorar si los datos son relevantes, extraer únicamente los datos relevantes o crear herramientas para que los especialistas obtengan información sobre estos datos [3].

Como se ha mencionado antes, la información digital no siempre se encuentra estructurada, por ejemplo se pueden encontrar escritas en leguaje natural. Esto supone una dificultad añadida al procesamiento de la información. Por ejemplo, extraer de una nota clínica todos los daños cerebrales, no es algo trivial, puesto que en el documento no aparece "Los daños cerebrales del paciente son..." sino que se desglosa según las distintas analíticas y en algunos de esos análisis pueden aparecer distintos problemas relacionados con el cerebro como por ejemplo "Traumatismo

craneal producido por...” o “Isquemia en arteria cerebral posterior...”. Por lo tanto parece lógico y necesario realizar un desglose del lenguaje para poder extraer información de las distintas palabras y conceptos, así como sus relaciones. El ámbito del procesamiento del lenguaje natural (NLP, Natural Language Processing) tiene como objetivo hacer desglose y extraer toda la información relevante. El NLP se apoya principalmente en las reglas sintácticas y semánticas, además de hacer un análisis léxico y morfológico de la lengua a procesar. Este campo es especialmente complejo puesto que cada idioma tiene reglas muy distintas y en su mayoría no tienen muchos elementos en común. Además no necesariamente tiene que ser un lenguaje correcto a nivel léxico, sintáctico o gramático, como puede ser un email o el contenido de las redes sociales lo que dificulta aún más la extracción de información.

Las técnicas de Machine Learning (o aprendizaje automático) son algoritmos que se utilizan principalmente para la clasificación, agrupamiento, asociación o predicción, utilizando un conjunto de características de ciertas instancias. Muchos de estos algoritmos se basan en reglas estocásticas, por lo tanto pueden llegar a extraer relaciones extremadamente complejas que a una persona podría llevarle años descubrir. Los ámbitos de aplicación de estos algoritmos son diversos y variados, siempre que una de las actividades anteriormente mencionadas esté presente, utilizar técnicas de aprendizaje automático va a ser una solución a contemplar. Como se ha mencionado en el párrafo anterior el lenguaje posee conceptos que están relacionados entre sí, por lo que estos algoritmos pueden ayudarnos a descubir esos vínculos. En el lenguaje natural se utilizan algunos elementos como pueden ser las siglas o los acrónimos, es decir, la creación de palabras a través de letras o silabas que componen el la palabra original, como por ejemplo “OMS” que significa Organización Mundial de la Salud. Sin embargo el uso de siglas y acrónimos crean ambigüedad, ya que podría ser cualquier concepto que tenga como primeras letras O, M y S. Un acrónimo o sigla tiene un concepto o definición asociando, este concepto se conoce como “expansión” de un acrónimo o sigla. En nuestro caso Organización Mundial de la Salud es la expansión de OMS. Durante la lectura de un texto expandir las siglas y los acrónimos no siempre es tarea fácil, una aproximación simple de como lo hace un ser humano sería: primero conocer posibles expansiones de acrónimo y segundo, analizar por el contexto del documento que esta expansión tiene sentido o guarda relación con el resto del documento o tiene una relación semántica con los elementos que lo rodean.

En este proyecto se va a tratar de aproximar esa metodología de desambiguación de acrónimos utilizando el procesamiento del lenguaje natural para extraer los elementos del texto y las técnicas de aprendizaje automático para localizar esas relaciones entre los conceptos. Todo esto enmarcado en un ámbito biomédico.

1.2 Motivación

Un campo de aplicación importante del NLP y Data Science es la medicina. Los textos clínicos son una fuente de información con varios rasgos a destacar: i) gran cantidad, tanto de historiales clínicos como de artículos de medicina, ii) constante crecimiento, lo que mejorará la precisión del sistema en el futuro, iii) alta fiabilidad, debido a la rigurosidad con que estos deben ser escritos, iv) estructuración parcial, a pesar de tratarse de documentos escritos en lenguaje natural, la rigurosidad con la que deben ser escritos seccionan el contenido de forma muy semejante, lo que facilita su análisis, v) la reciente predisposición de los hospitales a digitalizar los historiales clínicos.

Todas las características mencionadas hacen idónea la aplicación de técnicas de NLP sobre textos clínicos. El procesamiento del lenguaje natural en textos clínicos no es trivial en absoluto. A pesar de que el lenguaje médico intenta ser bastante preciso comparado con el lenguaje que usamos habitualmente, siempre quedan ambigüedades, abreviaciones o erratas que deben ser reconocidas por el sistema y no son fáciles de detectar e interpretar. En concreto el lenguaje médico abusa en especial de las abreviaciones, por lo que la necesidad de la desambiguación de los acrónimos y siglas es fundamental para la correcta interpretación de estos textos.

1.3 Objetivos

En relación con lo mencionado previamente, el objetivo principal de este trabajo es el estudio y análisis de técnicas de aprendizaje automático que faciliten la desambiguación de las abreviaturas en notas clínicas en español, y por lo tanto establecer relaciones correctas con todas sus posibles acepciones.

Para la consecución de este objetivo se proponen los siguientes objetivos parciales:

- Realizar un estudio en profundidad de las abreviaturas en las notas clínicas en español, realizando el proceso de desambiguación manualmente y detectar los conceptos, patrones o elementos que han ayudado a ese proceso.
- Comprender y asimilar el funcionamiento del procesamiento del lenguaje natural en español y en concreto en el ámbito de la medicina, determinando las diferencias con el procesamiento de lenguaje natural sin un ámbito específico.
- Analizar y extraer características interesantes de las abreviaturas que permitan entrenar modelos de aprendizaje automático.

- Determinar un conjunto de algoritmos que funcionen adecuadamente con la problemática del trabajo basándonos en proyectos ajenos y estudiando sus propiedades estadísticas.
- Evaluar y analizar los resultados obtenidos para poder determinar el funcionamiento del algoritmo, comparado con otras casuísticas para poder mejorar el rendimiento del mismo.

1.4 Estructura del trabajo

El presente trabajo consta de las siguientes secciones:

- En la segunda sección se describen y detallan todos los trabajos previos, relacionados con la desambiguación de conceptos media automatizada, mediante UMLS, técnica de Machine Learning o modelos de espacio vectorial. Adicionalmente, también se presentan los trabajos previos relacionados con Historia Clínica Digital.
- En la tercera sección se detalla el planteamiento del problema, contextualizando el marco teórico del mismo, así como detallando las herramientas usadas para resolver el problema.
- En la cuarta sección, se plantea la solución llevada a cabo. Definiendo el esquema de resolución del problema y la integración con el proyecto H2A.También se detallarán todas las aproximaciones realizadas para la resolución del problema. Y se concluirá con un análisis y comparación de los resultados obtenidos.
- Finalmente, en la quinta sección se explicarán las conclusiones del trabajo además de las líneas futuras.

2 TRABAJOS PREVIOS

2.1 Proyectos relacionados con Historia Clínica Digital(EHR)

El presente TFG se enmarca como parte de la investigación realizada en el grupo MIDAS de la Universidad Politécnica de Madrid, en el sistema H2A que tiene como objetivo extraer información procedente de documentos digitales relacionados con la salud de los pacientes, en concreto de documentos de texto e imágenes. H2A está inspirado en la arquitectura del sistema cTakes[4] de la universidad de Harvard que se centra en la extracción de conocimiento procedente del procesamiento de los documentos médicos de los hospitales. En este proyecto se hace un procesamiento de documentos en inglés por lo que el proyecto H2A explota la carencia de un sistema de este tipo en español y además ampliándolo con el análisis de imágenes. Como se menciona en Savova et al.[4], no hay recursos de uso público en el ámbito biomédico, como puede ser un corpus anotado. Por lo que para la evaluación de herramientas como cTAKES, se tuvo que desarrollar un corpus propio siguiendo los estándares de anotación lingüística de Penn TreeBank (PTB) [5] y el corpus GENIA[6] para poder entrenar los modelos. cTAKES procesa notas clínicas escritas en inglés identificando los conceptos biomédicos que aparecen, usando como referencia múltiples diccionarios agrupados en Unified Medical Language System (UMLS) que contiene conceptos como medicamientos, enfermedades sintomas y procedimientos médicos. Además se utiliza la herramienta UIMA para crear metadatos en las notas clínicas. Ambas herramientas han sido utilizadas en el proyecto H2A, sin embargo los diccionarios de UMLS en castellano son muy limitados y la cantidad de conceptos en este idioma en muy inferior en comparación con la cantidad de conceptos en inglés.

2.2 Relaciones conceptuales de UMLS

Algunos trabajos, como el de [7] aprovechan en uso de la herramienta UMLS para poder inducir la expansión de los conceptos en base a las relaciones jerárquicas de padres y hermanos de los distintos significados de la abreviación. Se centra en el intentar comprobar el grado de relación que tiene los conceptos que rodean la abreviatura con las distintas expansiones del acrónimo, seleccionando la que más grado de parentesco tenga con estos conceptos, por ejemplo “RA” solo tiene dos significados posibles “rheumatoid arthritis” y “renal artery” , entonces si los conceptos que rodean “RA” tienen más relación en la jerarquía de UMLS con la expansión “renal artery” esta será seleccionada como expansión correcta, sin embargo si “rheumatoid arthritis” está más relacionada, esta será la expansión seleccionada. A pesar de demostrar resultados bastantes satisfactorios, estos son debidos a que esta aproximación da mejores resultados que las técnicas de aprendizaje automático, siempre

y cuando las abreviaciones tengan pocas definiciones. Además la alta dependencia en UMLS tampoco es conveniente. Finalmente se descartó esta opción porque en número de conceptos de español de UMLS es muy reducido comparado con los conceptos en inglés.

2.3 Modelo de espacio vectorial

Otros proyectos como por ejemplo el de [8] utilizan los modelos de espacio vectoriales en el que extraer documentos con vectores de contexto y matrices de ocurrencia similares en notas clínicas en Sueco. Con esto se puede intentar inducir el significado que tienen los acrónimos en esos documentos ya que estos poseen una alta similitud. Sin embargo esta solución tiene el problema de una alta dimensionalidad en estos vectores de contexto y matrices de ocurrencia.

2.4 Desambiguación de acrónimos con técnicas de Machine Learning

En los trabajos [9][10][11][12][13] se usan soluciones basadas en técnicas de aprendizaje automático obteniendo resultados con un F1-score por encima del 90% en la mayoría de los casos, todos ellos en inglés, a excepción del [13] que se hace sobre textos en español y catalán. Las técnicas más comúnmente utilizadas son Naïve Bayes, Árboles de Clasificación y máquina de vector de soporte (SVM , Support Vector Machines) todas estas técnicas funcionan excepcionalmente bien en tareas de clasificación. Puesto que el problema de la desambiguación de abreviaciones se plantea como un problema de clasificación donde el acrónimo se clasifica en una expansión. Se han estudiado y explorado los bosques aleatorios, una técnica no utilizada para esta problemática, que sin embargo por sus características se ha decidido utilizar en este trabajo.

En [9] se demuestra que con pocos acrónimos y con pocos documentos, los modelos de espacio vectorial funcionan mejor que los algoritmos de Machine Learning, sin embargo en este trabajo se tiene un corpus de más de 30.000 datos además de contar con una generación de corpus completamente distinta.

Como se menciona en [10] algunas técnicas semisupervisadas también funcionan de forma más optima para la desambiaguación con pocos acrónimos y documentos, sin embargo hay que mencionar que se coincide con el uso de corpus externos procedentes de internet(Word Wide Web) para la creación del corpus anotado en este trabajo, en concreto se ha utilizado la página www.sedom.es para extraer las posibles definiciones de los acrónimos.

Los algoritmos de SVM, Naïve Bayes y árboles de clasificación ya han sido utilizados para la tarea de desambiguación en [11]. Además se utilizan características

como el PoS y n-gramas de ventana flexible demostrándose sus óptimos resultados. En este trabajo se utilizan características similares y algoritmos similares, aunque se han añadido más características y otros algoritmos. Sin embargo en este trabajo se realiza otra aproximación con el uso de n-gramas de ventana fija para poder ver la mejoría que pueden introducir otras características.

En [12] se ha estudiado el uso de Kernel Methods para reducir la dependencia del corpus etiquetado de forma manual al mínimo. Para ello utilizan texto sin etiquetar junto con ontologías semánticas que complementan el SVM con las posibles definiciones y sus diferencias.

3 PLANTEAMIENTO DEL PROBLEMA

3.1 Marco Teórico

Como se ha descrito previamente el objetivo principal de este trabajo es utilizar algoritmos de Machine Learning para llevar acabo la desambiguación automática de los acrónimos que aparecen en las notas clínicas. La ambigüedad que introducen los acrónimos y siglas presenta un gran problema para entender textos médicos donde se usan las mismas siglas en distintos contextos. Dentro del proyecto H2A, se enmarca un proceso de análisis de textos[14]. Dentro de este proceso existe una etapa de reconocimiento de conceptos médicos y establecer relaciones entre ellos. 1. Muchos de estos acrónimos ya existen en UMLS, pero debido que se está

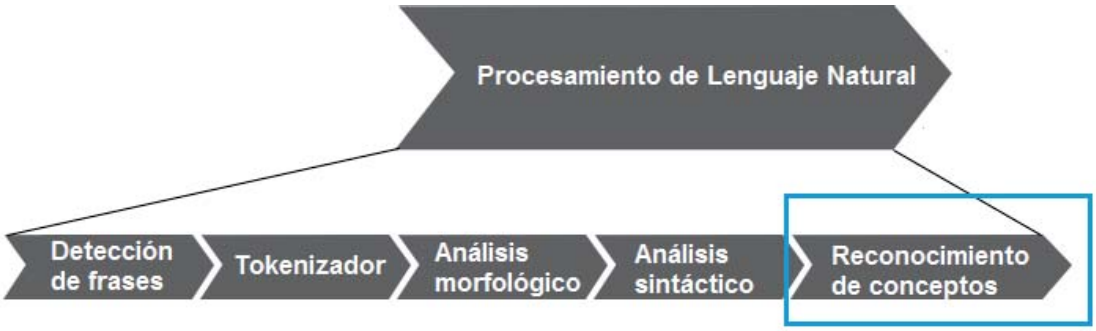

Figura 1: Proceso de análisis de textos del proyecto H2A

lidiando con textos en español la cantidad de acrónimos con lo que cuenta UMLS no son suficientes, por ello en este trabajo se ha decidido ampliarlo utilizando los acrónimos y las definiciones que aparecen en el diccionario de acrónimos médicos de www.sedom.es. El problema fundamental es que muchos de estos acrónimos tienen asociados múltiples definiciones que en diversas ocasiones son sinónimos. Para poder hacer este reconocimiento de conceptos de forma exitosa se necesita seleccionar la definición adecuada.

La tarea de desambiguar los acrónimos médicos se puede complicar incluso para los expertos, generalmente toda la información que se requiere para desambiguar se encuentra alrededor del acrónimo, es decir, se utiliza el contexto para poder llevar acabo la tarea. Por ejemplo, el acrónimo *"PCR"* tiene múltiples definiciones como pueden ser *"Proteína C Reactiva"* , *"Plantar Cutaneous Reflex"*, *"Polimerase Chain Reaction"*. En este caso dos de las tres definiciones están en inglés pero ambas suelen ser utilizadas comúnmente con esa acepción en las notas clínicas, en este caso determinar la sección en la que se encuentra suele ser crucial para determinar la definición adecuada puesto que es una de las arterias que se estudia en concreto

en esa prueba.*"Proteína C Reactiva"* suele aparecer cuando se está hablando de un análisis sanguíneo, mientras que *"Plantar Cutaneous Reflex"* puede aparecer en exploración general, sin embargo también se utiliza el acrónimo *"RCP"* para hacer referencia a esta definición. Finalmente la *"PCR"* aparece como *"Polimerase Chain Reaction"* cuando se esta hablando de microbiología como pueden ser los hemogramas. Con este ejemplo puede parecer que la tarea de desambiguar carece de la complejidad suficiente para poder aplicar técnicas de Machine Learning, sin embargo existen algunas relaciones no triviales y relaciones que un ser humano no es capaz de detectar, además de en muchas ocasiones al tratarse de un campo tan especializado como es el biomédico, es frecuente requerir de la experiencia de un experto para realizar la desambiguación. Este es el caso del acrónimo *"ACP"*, cuyas definiciones más comunes son *"Arteria Cerebral Posterior"* o *"Arteria Comunicante Posterior"*, ambas arterias se encuentran muy próximas, pertenecen al polígono de Willis y ambas pueden producir una isquemia. En este caso un experto al leer en el contexto y saber que se trata de un comentario procedente de un Duplex Transcraneal detecta claramente que se trata de *"Arteria Cerebral Posterior"* puesto que es una de las arterias que se estudia en concreto en esa prueba.

Por todo lo mencionado anteriormente se ha decido utilizar algoritmos de Machine Learning para resolver este problema como una tarea de clasificación, en la que se define si una acepción con unas características que serán extraídas previamente se corresponde con ese acrónimo o no. En este trabajo se han introducido características como palabras cercanas, conceptos médicos en la misma frase o la sección en la que se encuentra entre otros. Con toda esta información etiquetada previamente de forma manual se creará un modelo que sea capaz de determinar ante un caso nuevo la acepción correcta para un acrónimo. Por consiguiente una de las claves del éxito de este trabajo es determinar unas características influyentes y determinar el impacto que estas tienen sobre la tarea de clasificación.

3.2 Herramientas

En este apartado se describen y explican las herramientas utilizadas para llevar a cabo la desambiguación de acrónimos biomédicos.

3.2.1 UIMA Java Framework

Apache UIMA es una implementación de código abierto con licencia Apache que sigue la especificación de UIMA descrita en [15]. Ya que en este trabajo se está realizando un procesamiento de grandes volúmenes de información no estructurada la herramienta nos ofrece un gran número de facilidades a la hora de realizar este análisis. Con esta herramienta y con el uso de expresiones regulares se pueden de-

tectar elementos como las frases, las secciones que hay en una nota clínica los conceptos biomédicos, unidades de medida, entre otros. Esto resulta de gran utilidad para otorgarle a un fichero de texto plano información adicional, también conocida como metadatos o anotaciones, a determinadas palabras del mismo. Estas anotaciones o metadatos pueden ser utilizadas en fases posteriores del análisis del documento, evitando hacer de nuevo esa extracción de información. No solo nos permite tener esta información disponible en todas las fases del análisis sino que además tiene herramientas que facilitan el manejo de estos metadatos. Una misma palabra puede tener anotaciones de distintos tipos, por lo que solo dependiendo de el tipo de metadatos requeridos se busca en determinadas partes del documento. Un ejemplo simple de los distintos tipos de anotaciones puede ser; la palabra *"depresible"* puede tener una anotación que indica que esta en la frase número 5 y además otra anotación que indica que es un concepto biomédico que aparece en los repositorios de UMLS y toda la información de este concepto. Todos estos metadatos se almacenan en un fichero XML indicando para cada anotación de cada tipo el punto del fichero en el que empieza la anotación y el punto del documento en el que acaba. Este fichero XML que contiene todas las anotaciones así como el texto tiene una extensión XMI.

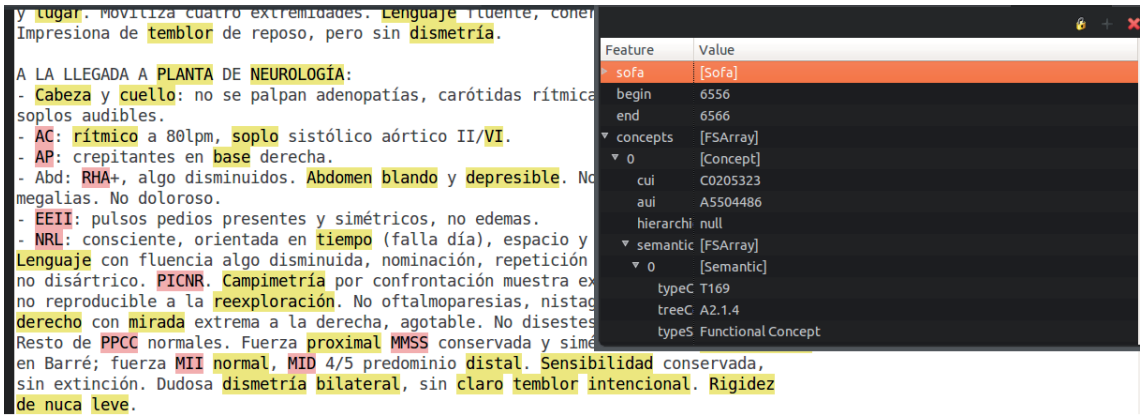

Figura 2: Ejemplo de anotación de concepto biomédico de la palabra "depresible"

Estas anotaciones pueden proceder de distintos documentos, el documento del que procede esa anotación se conoce como *"SofA"* (Subject of Analysis) que puede ser una utilidad muy interesante para relacionar conceptos de distintos documentos y generar anotaciones con información relevante. Finalmente cabe destacar que la herramienta UIMA ha sido de vital importancia a la hora de extraer las características a utilizar en los algoritmos de Machine Learning, así como crear nuevas anotaciones para la las definiciones seleccionadas para los acrónimos y poder integrar esta tarea de desambiguación en el proyecto h2a.

3.2.2 UMLS

UMLS (Unified Medical Language System) es un conjunto de ficheros y software que agrupa y relaciona gran parte del vocabulario, médico y biomédico aportando además estándares para permitir la interoperabilidad entre sistemas informáticos, en resumen, es una ontología de conceptos biomédicos.En concreto la parte que contiene la base de datos relacional se llama *Metathesaurus*. Este *Metathesaurus* es una base de datos multilingüe y multipropósito que contiene información de conceptos relacionados con la biomedicina y la salud, sus relaciones y sus posibles nombres entre otros. El *Metathesaurus* utiliza distintos vocabularios fuente, de donde se ha extraído la información sobre los conceptos. Con toda la información de estos vocabularios fuente se ha creado una jeraquía que relaciona estos conceptos y sus clasificaciones. Algunos vocabularios fuente que utiliza UMLS son: LOINC, SNOMEDCT y RxNORM. No solo utiliza estos vocabulario fuente como única fuente de información sino que también está enlazada con otras fuentes como la red semántica de UMLS y el léxico de SPECIALIST

El *Metathesaurus* esta organizado por conceptos o significados. En esencia esto vincula los posibles nombres o vistas del mismo concepto e identifica relaciones útiles entre los conceptos.

Algunas conceptos útiles que conviene mencionar sobre UMLS son:

- **Identificador Único de Concepto (CUI)**: Un concepto es un significado. Un significado puede tener múltiples nombres. Con esto se consigue relacionar todos los nombres que puede tener un significado (sinónimos) y unificar todos los nombres de las distintas fuentes en un único concepto.El CUI contiene la letra C seguida de siete números, por ejemplo C0018681.

- **Identificador Único léxico (término) (LUI)**: El LUI permite vincular todas las variantes léxicas de un nombre, estas variantes léxicas se detectan con el programa LVG (Lexical Variant Generator), que pertenece al conjunto de herramientas léxicas de UMLS. EL LUI contiene la letra L seguida de siete número.

- **Identificador Único de Nombre (SUI)**: Cada nombre de concepto único en cada idioma en el *Metathesaurus* tiene un SUI. Cualquier variación de este ya sea por mayúsculas u otros símbolos de puntuación es un nombre distinto por lo que tiene un SUI diferente.Un SUI contiene la letra S seguida de siete números.

- **Identificador Único de Átomos (AUI)**: Un átomo es un identificador de ocurrencia para cada cadena de caracteres única o concepto en las distintas fuentes de UMLS. Por lo tanto si la misma cadena de caracteres aparece varias

veces en el mismo vocabulario, como un nombre alternativo para diferentes conceptos un AUI único se le asigna a cada ocurrencia. Un AUI contiene la letra A seguido de siete números.

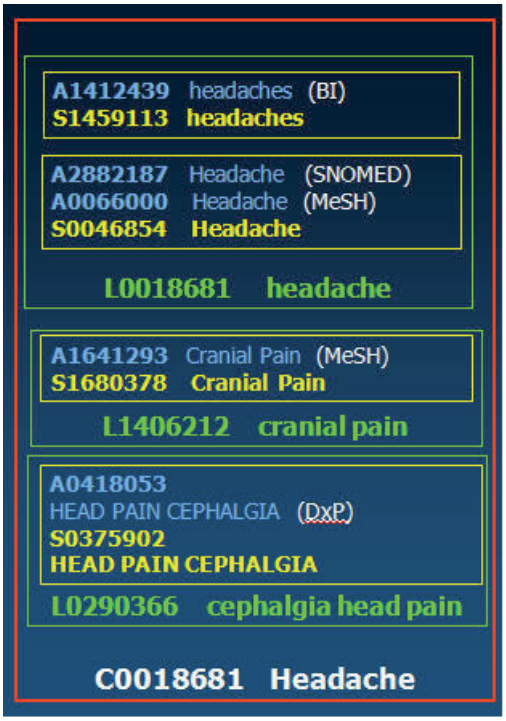

Figura 3: Ejemplo ilustrativo de la jerarquía de UMLS

Ya que todos los conceptos son asignado a al menos un tipo semántico de la red semántica de UMLS, esto otorga una categorización consistente de todos los conceptos en el *Metathesaurus*.

La herramienta UMLS ha resultado muy útil para la extracción de características para los algoritmos de Machine Learning.Además de ser utilizada en otras etapas del proyecto h2a.

3.2.3 Weka

Weka es una herramienta hecha en el lenguaje de programación Java, que agrupa diversos algoritmos de Machine Learning para tareas de minería de datos, por lo que ofrece herramientas que facilitan el preprocesado de los datos y el entrenamiento de los modelos para tareas de clasificación, regresión, clustering, entre otros. Adicionalmente, Weka ofrece algunas técnicas de validación de modelos que serán útiles en la etapa de evaluación de los modelos. Weka tiene una interfaz gráfica pero debido

a que el número de datos se ha optado por utilizar esta herramienta por linea de comandos. El gran volumen de datos y la complejidad de los mismo no solo ha afectado a como usar la herramienta, sino que también se ha requerido el uso de máquinas potentes puesto que estos algoritmos requieren de una gran capacidad de cómputo.

En concreto esta herramienta ha sido crucial para la fase de entrenamiento de modelos y la validación de los mismos. Al tener ya implementadas muchas de las técnicas de Machine Learning, ha agilizado este proceso de forma considerable.

4 SOLUCIÓN

4.1 Esquema de solución del problema

La figura 4 muestra el esquema de la solución que proponemos en este TFG para desambiguar acronimos y aprender modelos que en un futuro sirvan para anotar las historias clinicas digitales desambiguando los acronomimos.

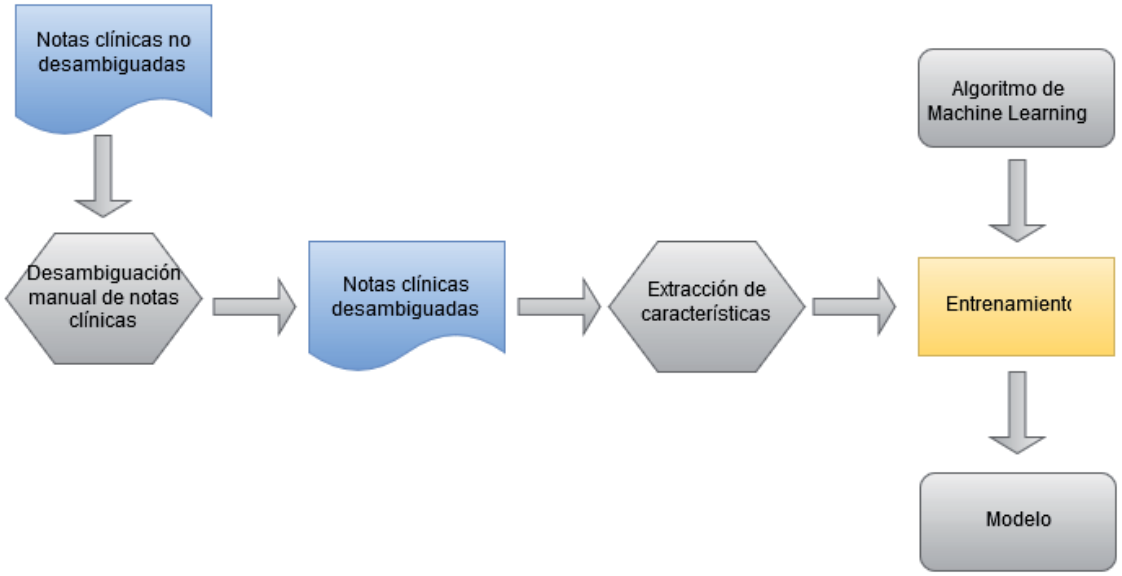

Figura 4: Diagrama del proceso seguido durante el trabajo

Como se puede observar en la figura 4 se desambiguará y modificará las anotaciones de los acrónimos de forma manual, ya que al utilizar técnicas de Machine Learning supervisadas, los datos introducidos al algoritmo requiere de un etiquetado previo de los datos. Después de desambiguar manualmente los documentos, se procederá a la extracción de características. Acto seguido, con las características extraídas, se introducirán todos estos datos al algoritmo de Machine Learning, esta fase viene descrita como entrenamiento. Finalmente se procederá con la evaluación del modelo. El proceso de etiquetado manual se ha realizado sobre 150 notas clínicas, con este gran número de documentos se genera una cantidad de información relevante y útil para poder entrenar los modelos y poder obtener resultado significativos.

4.2 Integración con el proyecto H2A

Tal y como se muestra en la figura 5 el proceso que proponemos se integra como pasos del proceso que actualmente se desarrolla en H2A. Nuestro proceso recibirá de H2A

un XMI donde los acrónimos no están desambiaguados. A continuación se extraerán las características descritas en la subsección 4.3 con estas características se ejecutará los modelos previamente entrenados y consecuentemente se tendrán resultados que se integraran en el XMI con las definiciones desambiguadas para cada acrónimo y esto se le pasara de nuevo a H2A para que siga el resto del procesamiento del texto.

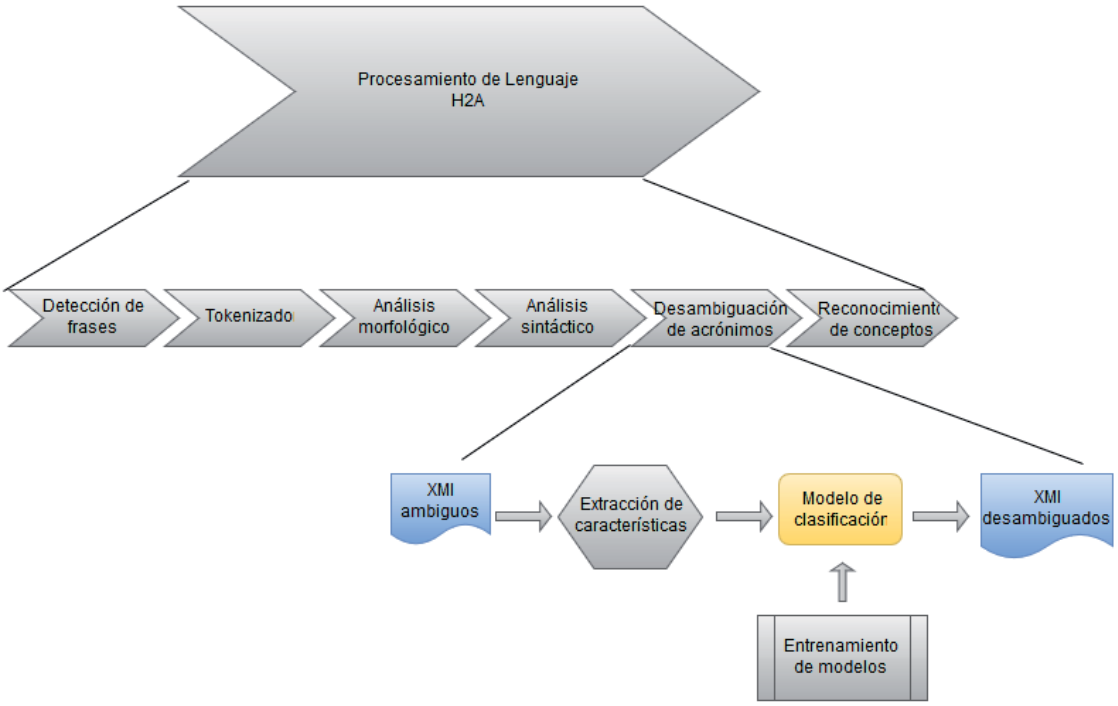

Figura 5: Integración con el proyecto H2A

4.2.1 Entrada del proceso de desambiguación

El proceso recibirá documentos XMI, que contendrá las anotaciones de acrónimos y todas las definiciones para este. Se le extraerán las características y se introducirá en el modelo.

4.2.2 Salida del proceso de desambiguación

Después de haber sido introducidas las características de cada acrónimo, el modelo seleccionará la definición adecuada y modificará las anotaciones. Por lo que finalmente la salida serán los ficheros XMI con las anotaciones de los acrónimos desambiguadas.

4.3 Anotaciones para la desambiguación

En este apartado se describirá como se han modificado las anotaciones de los acrónimos con UIMA y Java y también se explicará como se ha realizado la extracción de características con Java. Para el proceso de anotación de extracción de características se han utilizado 150 notas clínicas distintas.

4.3.1 Modificación de anotaciones

Para este trabajo se ha utilizado el anotador UIMA de acrónimos que ya estaba implementado, es decir, ya se estaba creado el tipo de anotación, el descriptor y el anotador. La anotación *AcronymAnnotation* envuelve a todo el acrónimo, y como información adicional se introduce en un String Array de UIMA todas las posibles definiciones de ese acrónimo. Es importante mencionar que las definiciones posibles proceden del repositorio de SEDOM. SEDOM (Sociedad Española de Documentación Médica) es un diccionario de siglas en español creado por la comunidad de médicos especializados. Todos los acrónimos de SEDOM han sido extraidos y almacenados en un documento JSON. El anotador realiza una búsqueda en los documentos de palabras que estén compuestas por varias letras en mayúscula, con puntos entre ellos o no. Así como letras sueltas en mayúscula. Solo se van a anotar aquellas palabras que en el etiquetado de Part of speech (PoS) o análisis morfológico se hayan anotado como nombres, es decir, que tengan la etiqueta *"NOUN"* en la anotación de PoS. De estos nombres solo se anotarán aquellos que estén formados por letras mayúsculas y tenga alguna coincidencia con los acrónimos extraídos de SEDOM. Finalmente si alguna palabra del texto cumple con los requisitos anteriores se crea una anotación con todas las posibles definiciones del acrónimo.

Para la desambiguación manual se eliminan todas las definiciones incorrectas para ese acrónimo en ese concepto dejándose solo la correcta. El objetivo del trabajo es crear un modelo que sea capaz de detectar cual de esta definiciones es la correcta y código java que integre el modelo y elimine todas las definiciones restantes. Este proceso de desambiguación estaría integrado dentro del pipeline de procesado de la arquitectura H2A, por lo que requiere leer el documento XMI, modificarlo y finalmente guardarlo dejando únicamente la definición adecuada de cada acrónimo.

4.4 Selección de algoritmos de Machine Learning

Para la selección de algoritmos de Machine Learning a utilizar, se ha realizado un estudio del problema de desambiguación, así como se han estudiado los algoritmos utilizados en otros proyectos del mismo tipo. Como novedad en este trabajo se han utilizado los Random Forest, que ofrecen ciertas ventajas a la hora de construir el modelo como se demuestra en [16].

4.5 Extracción de características para el entrenamiento

Para la selección de características se realizaron varias aproximaciones progresivas en las que se iban añadiendo y estudiando el impacto de estas características sobre la tarea de desambiguar. Las características fueron seleccionadas después de haber hecho una desambiguación manual, durante este proceso se realizó un análisis de metodología de la desambiguación que realiza el ser humano, así como un estudio de la estructura del documento. A continuación se describen las características extraídas en cada aproximación y el motivo por las que se han extraído dichas características.

4.5.1 Primera aproximación

Para la primera etapa del algoritmo, seleccionamos como características las palabras que están inmediatamente anterior y posterior al acrónimo para tratar de aprovechar la repetitividad que se observó en los textos, por lo tanto se hizo una aproximación semejante a la de los n-gramas. Por ejemplo la frase *"RCP extensor bilateral"* es una frase muy común y que además nos permite desambiguar como Reflejo cutáneo plantar de forma muy sencilla. Ya se ha demostrado la efectividad de estas características en [17] en el uso de algoritmos como los árboles de clasificación por lo tanto se va a usar también en este proyecto puesto que dos de los modelos utilizados son o están compuestos de árboles de clasificación.

Para la extracción de características ha sido necesario el uso del framework para Java de UIMA. Lo que se ha realizado ha sido una lectura de los XMI sobre las distintas anotaciones para extraer las características deseadas. En esta fase solo se ha hecho uso de las anotaciones de tipo *TokenAnnotation*, *SentenceAnnotation* y *AcronymAnnotation* que se detallan a continuación.

- **TokenAnnotation**: Es una anotación que detecta los *"tokens"* del texto, es decir, todos aquellos elementos del texto que pueden ser de interés. Los tipos que un *"tokens"* puede adquirir son: *"WORD"* si el token detectado es una palabra, *"PUNCTUATION"* si se trata de un signo de puntuación, *"NUMBER"* si el token detectado se trata de un número y *"SPECIAL"* para otros caracteres especiales. El *TokenAnnotation* ignora por completo todos los simbolos de espacio, tabulaciones y demás caracteres que no tienen una representación visual. Además esta anotación nos permite extraer el *"Lemma"*, o el conjunto de caracteres que conforman un token.

- **SentenceAnnotation**: Es una anotación que detecta las frases del documento del texto, almacenando el punto del texto en el que comienza y el punto del texto en el que acaba, así como numera de forma única dentro del documento cada frase.

- **AcronymAnnotation**: Es una anotación que detecta palabras que están conformadas por letras mayúsculas consecutivas o con elementos de puntuación entre medias y se ha encontrado coincidencias con el acrónimo en el JSON de SEDOM. En esta anotación se encuentra un *ArrayString* que contiene todas las definiciones posibles para este acrónimo. Obviamente, este anotador contiene la cadena de caracteres que componen el acrónimo.

Para extraer la información requerida se ha creado un programa Java que recorre mediante el uso de iteradores, provistos por el framework de UIMA para Java, los elementos de una determinada anotación. Por lo que con el iterador de *AcronymAnnotation* se han recorrido todos los acrónimos de los textos ya desambiguados, extrayendo de estos la cadena de caracteres que compone al acrónimo, así como el punto del fichero en el que empieza el acrónimo y el punto exacto al que acaba. Estos dos últimos valores son necesarios para detectar cuales son las palabras que hay justo antes y después del acrónimo. Lo que se realiza es un recorrido por todos los tokens que tiene *TokenAnnotation* hasta encontrar el token que pertenece al acrónimo. Posteriormente se desplaza el iterador una posición adelante y una posición hacia atrás. Con esto se han extraído las palabras anterior y posterior del acrónimo además del propio acrónimo. Sin embargo, parece lógico descartar aquellas palabras que pertenecen a otra frase. Este es el motivo por el que se usa el *SentenceAnnotation*. Así como se hizo con el *TokenAnnotation* en este anotador vamos a buscar la frase que envuelve al acrónimo y comprobar que el inicio y final de frase también envuelve a las palabras anterior y posterior. En caso de que la palabra anterior o posterior no esté en la misma frase se utilizarán los comodines *"BOS"* y *"EOS"*, que significan respectivamente *Begin of Sentence* y *End of Sentence*, en inglés, esto significa principio de frase y final de frase. Por lo tanto, si la palabra anterior no se encuentra dentro de la frase, se introducirá el comodín *BOS* en lugar de la palabra anterior, se actuará de forma análoga para el caso en el que la palabra posterior no pertenezca a a la frase.

Finalmente para la creación del dataset en la primera fase se cuenta con las siguientes características: ***preword*** que hace referencia a la palabra anterior al acrónimo, ***postword*** que hace referencia a la palabra posterior al acrónimo, ***Acronym*** que contiene la cadena de caracteres del acrónimo, ***area*** que representa el área médica a la que pertenece la nota clínica, en este caso es una característica irrelevante, pero tiene utilidad cuando se introduzcan notas clínicas de otros campos, actualmente esta característica siempre tiene el valor *"ICTUS"*, ***definition*** que contiene una de las posibles definiciones del acrónimo y finalmente la ***label*** que contiene los valores True o False dependiendo de si es la definición adecuada o no.

Para la creación del *dataset*, o fichero que contiene la información a introducir en el algoritmo de Machine Learning, se ha escrito por cada acrónimo n entradas,

donde n es el número de posibles definiciones que puede tomar ese acrónimo. Con este ejemplo se pretende ilustrar la información extraída por cada acrónimo. Para el texto:

"Fuerza proximal MMSS conservada y simétrica, sin claudicación en Barré; fuerza MII normal, MID 4/5 predominio distal."

preword,postword,acronym,area,definition,label
"proximal","conservada","MMSS",ICTUS,"Miembros superiores.",True
"fuerza","normal","MII",ICTUS,"Mamaria interna izquierda (arteria).",False
"fuerza","normal","MII",ICTUS,"Miembro inferior izquierdo.",True
","","4","MID",ICTUS,"Mamaria interna derecha (arteria).",False
","","4","MID",ICTUS,"Miembro inferior derecho.",True

El dataset generado es un fichero CSV (Comma Separated Value), para que la herramienta Weka lo pueda procesar adecuadamente, las palabras se encierran con el simbolo "" para que asuma que es una palabra completa y se utiliza el carácter coma para separar los distintos valores. Como se puede comprobar en el ejemplo anterior, la label True indica cual es la acepción correcta y la label False para la acepción incorrecta. En la primera aproximación se cuenta con un total cinco características exceptuando la característica de *label*.

4.5.2 Segunda aproximación

En la segunda fase se optó por ampliar el número de palabras alrededor del acrónimo, puesto que en ocasiones no son las palabras inmediatamente posteriores y anteriores las que nos ayudan a desambiguar los acrónimos, además de estar demostrado que aumentando la ventana de los n-gramas se consigue una mejor clasificación en [11]. El método de extracción de características es exactamente igual al de la primera fase 4.5.1. Los anotadores utilizados son los mismos. Lo único diferente es que cuando el iterador se sitúa encima del acrónimo se desplaza dos posiciones hacia atrás y posteriormente dos posiciones hacia adelante. Los comodines de BOS y EOS se utilizan de la misma forma que se utilizaban en la primera aproximación. Por lo tanto el dataset final queda igual que el anterior pero con dos campos más que son: ***prepreword*** que guarda la palabra que está dos posiciones por detrás del acrónimo y ***postpostword*** que contiene la palabra que se encuentra dos posiciones por delante del acrónimo. Resumiendo, el número de características extraídas hasta ahora alcanza el número de siete, sin contar con la característica label. Un ejemplo del dataset en la segunda aproximación podría ser:
Para el texto
"Fuerza proximal MMSS conservada y simétrica, sin claudicación en Barré; fuerza

MII normal, MID 4/5 predominio distal."

preword,postword,acronym,area,definition,prepreWord,postpostWord,label
"proximal","conservada","MMSS",ICTUS,"Miembros superiores.","Fuerza","y",True
"fuerza","normal","MII",ICTUS,"Mamaria interna izquierda (arteria).",";",",",False
"fuerza","normal","MII",ICTUS,"Miembro inferior izquierdo.",";",",",True
",","4","MID",ICTUS,"Mamaria interna derecha (arteria).","normal","/",False
",","4","MID",ICTUS,"Miembro inferior derecho.","normal","/",True

4.5.3 Tercera aproximación

Para la tercera fase se introdujeron las características de PoS de las palabras anteriores y el nombre de la sección en la que se ha encontrado el acrónimo. Como se ha mencionado anteriormente el PoS es el análisis morfológico de las palabras. La razón por la que se ha añadido el PoS es debido a que en muchas ocasiones saber que tipo morfológico tiene las palabras anterior y posterior ayuda mucho a la desambuguación de acrónimos. Por ejemplo es el caso de *"Rx. tórax AP y lat"* en este caso el acrónimo puede tomar muchas acepciones como *"Ausculatación pulmonar"*, *"Arteria pulmonar"*,*"Anteroposterior"* entre otros. En este caso la acepción más lógica y probable es que el acrónimo sea un adjetivo. En este caso efectivamente la acepción correcta era *"Anteroposterior"*. La sección ha resultado ser una de las características más útiles para desambiguar los conceptos. Por ejemplo, el acrónimo *"PCR"* tiene múltiples definiciones como pueden ser *"Proteína C Reactiva"* , *"Plantar Cutaneous Reflex"*, *"Polimerase Chain Reaction"*. En este caso determinar la sección en la que se encuentra suele ser crucial para determinar la definición adecuada puesto que *"Proteína C Reactiva"* y *"Polimerase Chain Reaction"* suelen ser las definiciones adecuadas cuando PCR se encuentra dentro de la sección *PRUEBAS*. Sin embargo cuando la sección se trata de *EXPLORACIÓN* la acepción correcta de *"PCR"* es *"Plantar Cutaneous Reflex"*, puesto que esta es una prueba muy común que se hace en el área de ICTUS para determinar el estado neurológico del paciente. En consecuencia las nuevas anotaciones que se recorren durante esta fase son:

- **PoS**: En esta anotación, se procesan todos los tokens detectados en *TokenAnnotation* y se clasifican según su tipo morfológico. Esta anotación clasifica los tokens en 9 tipos diferentes: *VERB* para verbos, *DET* para determinantes, *NOUN* para sustantivos, *ADP* para preposiciones, *ADJ* para adjetivos, *"."* para signos de puntuación, *PRON* para pronombres, *NUM* para números, *ADV* para adverbios.

- **RecordSection**: En esta anotación se detectan las distintas secciones en las

que está dividida una nota clínica. Se determinan el punto donde empieza y donde acaba la sección y se anota un nombre específico para las secciones. Los distintos tipos de secciones que se puede encontrar en una nota clínica son: *PRUEBAS* para las secciones que contienen todas las pruebas médicas realizadas al paciente, desde analíticas hasta resonancias magnéticas , *EXPLORACIÓN* en esta sección se encuentra toda la información relacionada con la exploración general realizada por el doctor a la llegada al hospital, *INGRESO* en la que se especifican los motivos por los que ingresa el paciente al hospital, *ANCEDENTES* donde se especifica los antecedentes médicos del paciente, *ENFERMEDAD* en esta sección se describe la enfermedad que padece el paciente con mayor calidad de detalles, *EVOLUCIÓN* en la que se describe la evolución del paciente desde el día de ingreso respecto a las medidas tomadas, *JUICIO CLÍNICO* en esta sección se especifica el diagnóstico final del doctor en base a la evolución del paciente y las pruebas realizadas y *TRATAMIENTO* donde se especifica el tratamiento a seguir por el paciente al alta del mismo.

El procedimiento realizado para extraer las características de PoS han sido similares a las de preword y postword. Se ha iterado por todos los PoS hasta llegar a la posición del acrónimo, se ha desplazado el iterador en ambas direcciones y se ha extraído el valor de PoS de ambas palabras. Para la extracción de la sección se ha seguido un procedimiento parecido al utilizado para determinar la frase a la que pertenece el acrónimo, pues ambas anotaciones contienen el intervalo de texto que engloban. Por lo tanto se ha iterado por las secciones hasta encontrar la que enmarca al acrónimo, es decir, que el valor inicial de la sección sea menor que el valor inicial del acrónimo y que además el valor final de la sección sea mayor que el valor final del acrónimo. Por lo tanto el número total de características es de diez si no tenemos en cuenta la *label*. Un ejemplo de dataset de la tercera aproximación para la frase:

"Fuerza proximal MMSS conservada y simétrica, sin claudicación en Barré; fuerza MII normal, MID 4/5 predominio distal."

... definition,prepreword,postpostword,section,prePoS,postPoS,label
..."Miembros superiores.","Fuerza","y","ENFERMEDAD","ADJ","ADJ",True
..."Mamaria interna izquierda (arteria).",";",",",","ENFERMEDAD","NOUN","ADJ",False
..."Miembro inferior izquierdo.",";",",",","ENFERMEDAD","NOUN","ADJ",True
..."Mamaria interna derecha (arteria).","normal","/","ENFERMEDAD",".","NUM",False
..."Miembro inferior derecho.","normal","/","ENFERMEDAD",".","NUM",True

4.5.4 Cuarta aproximación

Finalmente para la cuarta fase, debido a que en muchas ocasiones nos guiábamos de palabras que estaban alejadas del acrónimo pero pertenecían a la misma frase, se detectó que la mayoría de esas palabras, eran conceptos UMLS. Lógicamente conceptos médicos que pertenecen a la misma frase pueden ayudar a contextualizar al acrónimo. Se podría considerar un proceso similar al trabajo llevado a cabo en [7], en la que utilizan la jerarquía UMLS para determinar el parentesco de las definiciones de los acrónimos con los conceptos UMLS que rodean al acrónimo. Por todo lo mencionado anteriormente se decidió añadir los 6 conceptos UMLS más cercanos al acrónimo y que pertenezcan a la misma frase. Un claro ejemplo de la eficiencia de estas características es encontrar *"FA 48mm3"* en la mayoría de los casos se desambigua como *"Flutter Auricular"* o *"Fase Acelerada"*, puesto que este acrónimo es muy común en la sección de pruebas en la que se hace un TAC torácico o un Electrocardiograma. Sin embargo, los conceptos UMLS *"Bioquímica"*, *"mm3"*, *"Colesterol"* nos ayudan a determinar que en este caso FA se trata de *"Fosfata Alcalina"* que es una medida que no se suele remarcar de esta forma habitualmente en los resultados de las analíticas de sangre.

Por lo mencionado anteriormente el nuevo anotador a utilizar es:

- **Umls**: Este anotador se encarga de detectar aquellos tokens que se encuentran dentro de la base de datos de UMLS, añadiendo toda la información proporcionada por UMLS, como ejemplo se puede señalar la figura 3 que es un ejemplo de una anotación UMLS.

El proceso a seguir para realizar la extracción de estas características ha sido utilizar la frase del acrónimo, conocida desde la primera aproximación, y encontrar todos los conceptos UMLS dentro de la frase. Se han utilizado dos estructuras de datos distintas, una pila para la mitad de la lista de conceptos UMLS que se encuentran antes del acrónimo y una $LinkedList< String >$ de Java para los UMLS que se encuentran después del acrónimo. El motivo principal de el uso de estas dos estructuras de datos es para priorizar la escritura de los conceptos UMLS más cercanos al acrónimo, puesto que estos estarán más relacionados con el acrónimo que los más alejados. En la situación ideal se escribirán seis conceptos UMLS, tres anteriores al acrónimo y tres posteriores. Sin embargo si una de las dos mitades carece de estos tres elementos se compensa con los elementos sobrantes de la otra mitad, para evitar a toda costa la pérdida de estos conceptos. Por lo tanto si tenemos cuatro elementos anteriores y dos posteriores, uno de los elementos anteriores ocupará el lugar del concepto posterior que falta. Por desgracia muchos de estos acrónimos no contienen siempre seis conceptos UMLS dentro de la misma frase. Por lo tanto se utiliza el comodín *"NC"* (No concepto) para suplir los conceptos carentes en la frase

del acrónimo. Esto también puede suponer una información de gran interés puesto que igual ciertas definiciones nunca van acompañadas de otros conceptos UMLS. Por lo tanto el dataset final del trabajo se compone de dieciséis características sin contar la label. Para la frase:

"Fuerza proximal MMSS conservada y simétrica, sin claudicación en Barré; fuerza MII normal, MID 4/5 predominio distal."

...postPoS,umlsconcept1,umlsconcept2,...,umlsconcept6,label
..."ADJ","proximal","distal","normal","claudicación","NC","NC",True
..."ADJ","proximal","claudicación","distal","normal","NC","NC",False
..."ADJ","proximal","claudicación","distal","normal","NC","NC",True
..."NUM","proximal","claudicación","distal","normal","NC","NC",False
..."NUM","proximal","claudicación","distal","normal","NC","NC",True

4.5.5 Análisis de rendimiento de los algoritmos

Antes de comenzar, explicar que hay que entender el dataset como una matriz de x filas e y columnas, donde las filas son las muestras obtenidas y las columnas son las características de un objeto. Cabe destacar que el dataset ha sido modificado de tal forma que el orden de las columnas ha sido modificado aleatoriamente. Esta modificación se ha hecho para garantizar que en los distintos pasos de la técnica de validación cruzada se tienen siempre datos significativos y variados. De otra forma el dataset tendría el orden de aparición de los acrónimos dentro del XMI y habría secciones de dataset monopolizadas por un solo concepto.

En este apartado se desarrollará una comparación y estudio de los modelos creados. Para este trabajo se han creado modelos usando la herramienta Weka que contiene múltiples implementaciones de distintos algoritmos de Machine Learning utilizados para minería de datos. Los algoritmos seleccionados han sido: Support Vector Machine (SVM), Random Forest (RF) y Árboles de decisión C45. Todos estos algoritmos se explicarán en profundidad en secciones posteriores. Para estudiar el impacto de las características se han entrenado modelos del algoritmo de Árbol de decisión C45 debido a la rapidez de creación de modelos a causa de la baja complejidad de entrenamiento.

Para realizar la validación de los modelos se ha utilizado la técnica de validación cruzada, más comúnmente conocida como Cross-Validation. Esta técnica de validación consiste en dividir el dataset provisto al algoritmo en *n-folds*, es decir n subconjuntos independientes. El número de subconjuntos trata de estar balanceado, pero sin embargo para garantizar que la información que contienen estos subcon-

juntos sea los equilibrada posible, como se comenta anteriormente, se ha hecho un barajado de los datos a nivel de fila. Por lo tanto una vez dividido el dataset en n subconjuntos, se hacen n entrenamientos. Cada entrenamiento se hace con *n-1* subconjuntos, el subconjunto restante se utiliza como entrada al modelos para estudiar el comportamiento del mismo, es decir, se utiliza para probar el modelo. En cada iteración se va utilizando un subconjunto de prueba distinto por lo que cada iteración también cuenta con subconjuntos de entrenamiento distintos. Una vez hechos todos los entrenamientos y pruebas, el resultado final de las pruebas, es la media de todos los resultados intermedios. Con esto se puede garantizar que todo el dataset ha sido usado tanto para entrenamiento como para pruebas y por lo tanto se considera una forma válida de analizar el comportamiento del modelo. En la figura 6 se puede observa de forma más visual el método de validación cruzada.

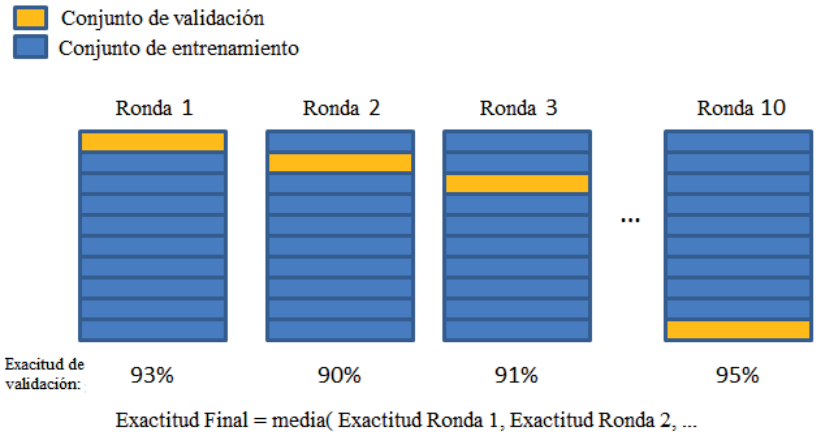

Figura 6: Ejemplo de cross validation 10 folds

Las medidas que se han tomado para comprobar el rendimiento del modelo han sido la *Precision* o precisión, que indica de los positivos detectados, cuantos eran realmente positivos, *Recall* o sensitividad, que indica el ratio de positivos detectados sobre los positivos existentes y finalmente el *F1-score* indica la exactitud del modelo utilizando las dos medidas anteriores, es una medida que te muestra de forma más rápida el rendimiento *"total"* del modelo. Estas son las fórmulas para calcular estas medidas:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{1}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{2}$$

$$F1score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3}$$

Donde:

- **True positives**: aquellos en los que el modelo clasifica como True y realmente eran True.
- **False positives**: aquellos en los que el modelo clasifica como False y realmente eran True.
- **True negatives**: aquellos en los que el modelo clasifica como False y realmente eran False.
- **False negatives**: aquellos en los que el modelo clasifica como False y realmente eran True.

Una forma más visual de analizar los aciertos o fallos de clasificación es la matriz de confusión. En esta matriz se ven los *True positives, False positives, True negatives* y *False negatives*.

4.6 Comparación teorica de algoritmos

4.6.1 Árboles de decisión J48

Los árboles de decisión J48 son la implementación utilizada por la herramienta Weka de los árboles de decisión C4.5 expuestos en [18]. Los árboles de decisión consiste en crear un árbol en la que se va navegando por los nodos en base a los valores que presentan las distintas características, siendo las hojas el resultado de la clasificación True o False para la definición de un determinado acrónimo. La construcción de estos árboles se basa en la teoría de la entropía de Shanon, es decir, la cantidad de información que aporta un acontecimiento. En este caso el acontecimiento es la característica. Se selecciona la característica con mayor entropía y se crean tantos nodos intermedios como valores o rangos de valores tome esa determinada característica. Este es un proceso iterativo que consume de una gran cantidad de memoria, pero tiene una complejidad de entrenamiento lineal, o en notación de O grande, $O(pn)$, donde n es el número de muestras del dataset y p el

número de características. Esto es debido a que en el peor de los casos el número de nodos máximo que podría haber es el número de muestras del dataset.

4.6.2 Random Forest

Los Random Forest son una técnica de Machine Learning que utiliza el concepto de *"bagging"*, es decir, es crear un modelo compuesto por varios modelos y que para tomar una decisión elige la opción más votada entre todos sus modelos. Este principio se basa en que todos los modelos tienen a acertar y que los modelos se equivocan en puntos distintos. En este caso un bosque aleatorio esta compuesto de múltiples árboles de clasificación. La razón principal para usar bosques aleatorios que resuelve el problema de compensación de bias-varianza que presentan los modelos. Esto es si el bias aumenta, la varianza se reduce y viceversa. El bias representa lo distante que esta un modelo de acertar, es decir, si lo viésemos como una diana, donde el centro es el acierto al 100%, el bias indica lo alejados que estamos del centros. La varianza representa la dispersión de las predicciones que se hacen. Los bosques aleatorios consiguen reducir la varianza de las predicciones sin aumentar apenas el bias. Los bosques aleatorios reciben este nombre ya que selecciona un subconjunto aleatorio de características de los datos introducidos, así como un subconjunto aleatorio de los datos introducidos, es decir, selecciona un número de columnas aleatorias y un numero de filas aleatorias para construir cada uno de los modelos de los que esta compuesto. Esta aleatoriedad consigue crear modelos relativamente distintos para tener variabilidad en la toma de decisiones, además con la selección aleatoria de características puede estar evitando que los modelos selecciones características irrelevantes. Otra gran ventaja de este algoritmo es que tiene una complejidad de entrenamiento relativamente baja ya que en notación de O grande podríamos definirla como: $O(MKNlog^2N)$ Donde: N = número de muestras K <= p,donde p es el número de características. M = el número de árboles en el bosque.

Además hay que tener en cuenta que como es una técnica de agrupamiento de modelos, el entrenamiento se hace sobre subconjunto aleatorio de datos y características, por lo tanto el número de muestras N va a ser de alrededor de un 63.2% de las muestras totales, como ha demostrado Gille Louppe en [16].

4.6.3 Support Vector Machines (SVM)

La Support Vector Machine trata de construir un hiperplano de alta o infinita dimensionalidad para poder separar las distintas clases que se encuentran en el dataset. Generalmente este tipo de problemas no es linealmente separable y se utilizan los denominados *kernels* para proyectar los puntos del dataset en un espacio de una dimensión mucho mayor. Sin embargo el uso de estos *kernel* afecta drásticamente al resultado de la clasificación. Por lo que hay que probar los más relevantes para

ver cual se adecua mejor al dataset. Un gran inconveniente de las máquinas de soporte vectorial es su complejidad de entrenamiento. Como se demuestra en [19], las máquinas de soporte vectorial escalan muy bien en número de características, pero escalan realmente mal cuanto mayor es el dataset. En notación de O grande, la complejidad de entrenamiento de las SVM es de O^3, donde n es el numero de muestras del dataset.

4.7 Experimentación

En las tablas 1,2 y 3 se puede observar las matrices de confusión de los distintos modelos.

C45		Clasificación	
		True	False
Realidad	True	8579	372
	False	605	19556

Tabla 1: Matriz de confusión del árbol C45

RF		Clasificación	
		True	False
Realidad	True	8473	478
	False	514	19647

Tabla 2: Matriz de confusión del Random Forest

SVM		Clasificación	
		True	False
Realidad	True	7663	1293
	False	1052	19114

Tabla 3: Matriz de confusión del SVM

Con esta tabla, ya se puede intuir los resultados de los modelos. Se ve una clara diferencia entre el árbol de decisión C45 y el Random Forest respecto a la máquina de soporte vectorial. Claramente el C45 y el bosque aleatorio aciertan más que la máquina de soporte vectorial. Y entre el árbol C45 y el Random Forest se puede observar que el árbol C45 clasifica mejor los casos positivos, mientras que el Random Forest clasifica mejor los casos negativos. Esto se puede observar en que,

en el bosque aleatorio, la tasa de False negatives es menor y la tasa de False positives es mayor y viceversa para el árbol de decisión C45.

En la tabla 4 se puede observar el cambio en el rendimiento del modelo según las características que vamos introduciendo. Como se ha mencionado anteriormente se usará el algoritmo de árbol de decisión C45 como referencia.

C45			
Medidas	Precision	Recall	F1-Score
Aproximación 1	0.963	0.963	0.963
Aproximación 2	0.964	0.963	0.963
Aproximación 3	0.967	0.966	0.966
Aproximación 4	0.967	0.967	0.967

Tabla 4: Tabla comparativa de características

Como se puede observar en la tabla el incremento de características se ve reflejado en las medidas tomadas del modelo. Como se puede observar donde se ha obtenido una mejora mayor ha sido en la aproximación tres, pues como ya se comenta en la sección 4.5.3 las características introducidas son de vital importancia para la desambiguación. Sin embargo, en la cuarta fase solo se ha mejorado ligeramente el recall. Esto puede ser debido a que el número de conceptos UMLS introducidos es muy elevado y en la mayoría de los casos aparecen más comodines que conceptos, es decir, es una característica útil pero muy situacional.

En la tabla 5 se puede observar una comparativa de rendimiento de los distintos algoritmos, que nos permite identificar al algoritmos que está realizando una mejor clasificación.

17 características			
Medidas	Precision	Recall	F1-Score
Árbol C45	0.967	0.966	0.967
Random Forest	0.966	0.966	0.966
SVM	0.919	0.920	0.919

Tabla 5: Tabla comparativa del rendimiento de los algoritmos

Como se puede observar en esta tabla, el algoritmo que ha resultado tener mejor rendimiento en la clasificación ha sido el árbol de decisión, también hay que destacar que modelo Random Forest, ha logrado conseguir un rendimiento casi equiparable al del árbol de decisión C45, por lo tanto queda demostrada su utilidad para la tarea de desambiguación. Por otra parte, el modelo SVM ha obtenido resultados muy

inferiores en comparación con los modelos anteriores, además de presentar el gran inconveniente de un gran tiempo de entrenamiento.

5 CONCLUSIONES

5.1 Conclusiones

En este TFG se ha presentado el diseño e implementación de un anotado basado en UIMA que permite desambiguar acrónimos en notas médicas. Para la realización de este desambiguador es necesario entrenar modelos de clasificación sobre textos anotados manualmente. En este trabajo se han mostrado el procedimiento de extracción de carácterísticas para el entrenamiento de los modelos, así como los resultados de la experimentación sobre una muestra de 150 notas clínicas en las que se han aplicado los algoritmos de árboles de decisión , SVM y Random Forest y donde los árboles de decisión han resultado ser los más eficientes sobre los datos utilizados para resolver el problema de desambiguación. El anotador se ha integrado correctamente dentro del proyecto H2A y consiguientemente todos los objetivos se han cubierto con éxito.

5.2 Líneas futuras

5.2.1 Selección de nuevos atributos

Para intentar mejorar el rendimiento del modelo actual, se plantea la introducción de nuevas características. Una de las características a introducir será, extraer más palabras anteriores y posteriores. Así mismo, aumentar también el número de PoS anteriores y posteriores. Para los conceptos UMLS que rodean al acrónimo dentro de la frase, va a tratar de expandirse a los conceptos UMLS dentro de una subsección, como por ejemplo, dentro de la sección *PRUEBAS* detectar la subsección *Análisis de sangre*.

5.2.2 Algoritmo de mejor rendimiento

Se explorarán otras técnicas que ya han demostrado su utilidad en el campo de la minería de datos como se ha estudiado en [20]. En concreto se van a probar los algoritmos de Naïbe Bayes, K-medias y KNN.

5.2.3 Preprocesado de datos

Para reducir el número de definiciones posibles, y por lo tanto reducir la ambigüedad de un acrónimo, se ha ideado un preprocesado de datos que consiste en: Recorrer todas las anotaciones de AcronymAnnotation y tratar de extraer el CUI de las definiciones (Véase 3.2.2) y con ello utilizar la jerarquía UMLS para reducir el número de definiciones posibles.

5.2.4 Desambiguación de conceptos biomédicos

Finalmente, una de las líneas futuras más prometedoras es trasladar todos los conocimientos adquiridos durante la realización de este trabajo y tratar de resolver el problema de la ambigüedad a nivel de concepto.

Bibliografía

[1] "Internet users in world by regions (2015) [online]. available: http://www.internetworldstats.com/stats.htm."

[2] T. economist, "[online].available: http://www.economist.com/node/15557443."

[3] G. Press, "(2013,may,28). a very short story of data science [online]. available: http://www.forbes.com/sites/gilpress/2013/05/28/a-very-short-history-of-data-science/fbb73d69fd23."

[4] G. K. Savova, J. J. Masanz, P. V. Ogren, J. Zheng, S. Sohn, K. C. Kipper-Schuler, and C. G. Chute, "Mayo clinical text analysis and knowledge extraction system (ctakes): architecture, component evaluation and applications," *Journal of the American Medical Informatics Association*, vol. 17, no. 5, pp. 507–513, 2010.

[5] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of english: The penn treebank," *Computational linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[6] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, "Genia corpus?a semantically annotated corpus for bio-textmining," *Bioinformatics*, vol. 19, no. suppl 1, pp. i180–i182, 2003.

[7] H. Liu, S. B. Johnson, and C. Friedman, "Automatic resolution of ambiguous terms based on machine learning and conceptual relations in the umls," *Journal of the American Medical Informatics Association*, vol. 9, no. 6, pp. 621–636, 2002.

[8] L. Tengstrand, B. Megyesi, A. Henriksson, M. Duneld, and M. Kvist, "Eacl-expansion of abbreviations in clinical text," in *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)@ EACL*, pp. 94–103, 2014.

[9] M. Stevenson, Y. Guo, A. Al Amri, and R. Gaizauskas, "Disambiguation of biomedical abbreviations," in *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pp. 71–79, Association for Computational Linguistics, 2009.

[10] S. Pakhomov, T. Pedersen, and C. G. Chute, "Abbreviation and acronym disambiguation in clinical discourse," in *AMIA Annual Symposium Proceedings*, vol. 2005, p. 589, American Medical Informatics Association, 2005.

[11] M. Joshi, S. V. Pakhomov, T. Pedersen, and C. G. Chute, "A comparative study of supervised learning as applied to acronym expansion in clinical reports.," in *AMIA*, 2006.

[12] X. Li, S. Qing, H. Zhang, T. Wang, and H. Yang, "Kernel methods for word sense disambiguation," *Artificial Intelligence Review*, pp. 1–18, 2015.

[13] G. E. Bakx, L. Villodre, and G. Claramunt, "Machine learning techniques for word sense disambiguation," *Unpublished doctoral dissertation, Universitat Politecnica de Catalunya*, 2006.

[14] R. Costumero, A. Garcia-Pedrero, I. Sánchez, C. Gonzalo, and E. Menasalvas, "1 electronic health records analytics: Natural language processing and image annotation," *Big Data and Applications*, p. 1, 2014.

[15] D. Ferrucci, A. Lally, K. Verspoor, and E. Nyberg, "Unstructured information management architecture (uima) version 1.0," 2008.

[16] G. Louppe, "Understanding random forests: From theory to practice," *arXiv preprint arXiv:1407.7502*, 2014.

[17] T. Pedersen, "A decision tree of bigrams is an accurate predictor of word sense," in *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pp. 1–8, Association for Computational Linguistics, 2001.

[18] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[19] A. Abdiansah and R. Wardoyo, "Time complexity analysis of support vector machines (svm) in libsvm," *International Journal Computer and Application*, 2015.

[20] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip, *et al.*, "Top 10 algorithms in data mining," *Knowledge and information systems*, vol. 14, no. 1, pp. 1–37, 2008.

Este documento esta firmado por

Firmante	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
Fecha/Hora	Mon Jun 06 19:39:51 CEST 2016
Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
Numero de Serie	630
Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)