Universidad de Lima

Facultad de Ingeniería y Arquitectura

Carrera de Ingeniería de Sistemas

SISTEMA DE DIGITALIZACIÓN Y ESTRUCTURACIÓN DE INFORMACIÓN CLÍNICA CON TÉCNICAS DE RECONOCIMIENTO ÓPTICO DE CARACTERES Y PROCESAMIENTO DEL LENGUAJE NATURAL

Trabajo de investigación para optar el Título Profesional de Ingeniero de Sistemas

Hugo Eduardo Castro Aranzábal

Código 20120293

Walter Giancarlo Pinedo Barrientos

Código 20122082

Asesor

Juan Manuel Gutiérrez Cárdenas

Lima – Perú

Marzo del 2018

UNIVERSIDAD DE LIMA
SCIENTIA ET PRAXIS

SISTEMA DE DIGITALIZACIÓN Y ESTRUCTURACIÓN DE INFORMACIÓN CLÍNICA CON TÉCNICAS DE RECONOCIMIENTO ÓPTICO DE CARACTERES Y PROCESAMIENTO DEL LENGUAJE NATURAL

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

El presente trabajo busca desarrollar un sistema que permita tanto la digitalización como la estructuración de registros clínicos a partir de apuntes escritos por el doctor de forma tradicional, resultando en un proceso que no resulte intrusivo a su flujo de trabajo. Por lo tanto, la solución permitirá portar la información del manuscrito desde un medio físico, como el papel, hacia un medio de almacenamiento digital, como un repositorio de datos, que contenga las enfermedades categorizándolas contra la décima edición de la clasificación internacional de enfermedades (CIE-10).

Para ello, se aplicaron distintas técnicas tradicionales de visión por computador para realizar el preprocesamiento como dilatación, erosión, normalizado, entre otras, sobre imágenes obtenidas de la base de datos NIST SD19 las cuales fueron utilizadas por el algoritmo k-nearest neighbors para que se pueda realizar el reconocimiento óptico de caracteres, además se hizo una optimización del número de vecinos cercanos junto con otras mejoras que permitieron obtener una sensibilidad más alta.

Posteriormente, se aplicaron algunas técnicas para el procesamiento del lenguaje natural como el POS-tagging, chunking, eliminación de stop words, stemming, TF-IDF, entre otros para luego realizar la categorización de enfermedades según la CIE-10 se utilizó un Support Vector Machines. Finalmente, el resultado permitirá tener a los registros clínicos digitalizados con su respectivo código de enfermedad para luego ser almacenados y de esta forma se tendrá la disponibilidad de estos datos para futuros proyectos.

ABSTRACT

The following work aims to develop a system that allows both the digitization and the structuring of clinical records from notes written by a doctor in a traditional way, resulting in a process that is non-intrusive to their workflow. Therefore, the solution will allow carrying the manuscript information from a physical medium, such as a piece of paper, to a digital storage medium, such as a data repository, containing different diseases categorizing them against the tenth edition of the international classification of diseases (ICD-10).

To do this, different traditional computer vision techniques were applied to perform preprocessing, such as dilating, eroding, normalization, among others, on images obtained from the NIST SD19 database, which were used by the k-nearest neighbors algorithm to perform the optical character recognition. In addition, an optimization of the number of nearest neighbors was made along with other improvements that allowed to obtain a higher sensitivity.

Subsequently, some techniques were applied to the processing of natural language such as POS-tagging, chunking, stop words filtering, stemming, TF-IDF, among others, to later categorize diseases according to the ICD-10 using a Support Vector Machines. Finally, the result will allow to have the clinical records digitized with their respective disease code and then stored, making these data available for future projects.

INTRODUCCIÓN

En los últimos años, los avances tecnológicos han generado diversos giros en la realidad de muchos países. Las empresas que fabrican productos electrónicos han ido marcando la pauta en los últimos avances que buscan facilitar la rutina diaria de una persona como son los teléfonos inteligentes, nuevas plataformas de entretenimiento, entre otros (La República, 2010). Estos avances se han visto reflejados en trabajos interdisciplinarios en las más importantes industrias como agricultura, minería, educación, entretenimiento, salud u otros sectores cuya importancia y priorización depende del plan y políticas de cada país.

En el caso del sector salud, poseer tecnología de gran calidad y lo más accesible posible para la población es uno de los principales desafíos que tienen los gobiernos (Ayala, E. et al, 2007). No cabe duda alguna que actualmente la sociedad se encuentra en una revolución digital que llama a la transición a la tecnología digital en todos los campos.

La telemedicina, registros clínicos electrónicos, sistemas de integración entre diferentes clínicas, aplicaciones móviles que velan por el paciente son algunos de los resultados de dicha revolución en los últimos años. Como consecuencia, la información ha ido incrementado y con ello las técnicas de análisis se han vuelto más sofisticadas para lograr un mejor entendimiento de estos datos.

La realidad en el Perú no es distinta a la que existe cruzando las fronteras, si bien la revolución que se viene dando no es tan ágil como en otros países, existen avances que se están impulsado en base a leyes. Para el alcance de la presente investigación, un claro ejemplo es la Ley 30024, dictada en el 2013 como Ley que crea el Registro Nacional de Historias Clínicas Electrónicas (Ministerio de Salud, 2013).

Dicha ley promueve la digitalización de las historias clínicas garantizando la protección de datos en el proceso de atención y servicios médicos de apoyo que implementan historias clínicas electrónicas. Además, optimizaría el uso de recursos y reducir la duplicidad de procedimiento (El Peruano, 2015).

Sin embargo, aun cuando la ley ya fue dada en el decreto N° 039-2015-SA, existe una transición que a julio del 2017 no se ve mayores avances. Algunas clínicas ya han optado por utilizar registros clínicos digitales mediante un sistema donde los doctores ingresan la información al sistema. Sin embargo, no todos los centros de salud poseen muchas veces los recursos para costear un equipo informático y soportar este tipo de soluciones.

En este escenario, los doctores aún siguen utilizando el lapicero y papel como herramientas para entregar el diagnóstico a sus pacientes, y toda la información generada en estas instituciones también es relevante tanto para el paciente como para la investigación científica del campo (Hilbert, M., 2015). Por lo que tener esta información no digitalizada se hace difícil poder analizarla para el beneficio de la institución o su efectiva administración.

Ante la problemática expuesta anteriormente, el presente trabajo de investigación busca desarrollar un sistema que permita a las instituciones clínicas convertir los datos no estructurados de los registros clínicos manuscritos a estructurados para luego almacenarlos en una base de datos. Esto se logrará mediante técnicas de reconocimiento óptico de caracteres (OCR) y procesamiento del lenguaje natural (NLP).

DESCRIPTORES TEMÁTICOS

- Optical Character Recognition
- Electronic Health Record
- Healthcare Information Technology
- International Statistical Classification of Diseases
- Natural Language Processing
- Term frequency – Inverse document frequency
- Support Vector Machine

CAPÍTULO I: PLANTEAMIENTO DEL PROBLEMA

1.1 Formulación del problema

La sociedad actualmente se encuentra bastante influenciada por las tecnologías de información y comunicaciones, donde el sector salud no es excepción a ello. Debido a ello, en distintas partes del mundo se ve la adopción de sistemas de registros electrónicos. En algunos casos, como en EEUU, el gobierno promueve en gran medida la adopción de estos sistemas, (Simborg, 2008) con lo que llegan a alcanzar tasas de adopción de 96.4% entre hospitales (Charles, Meghan, & Searcy, 2015) y esta cifra es muy similar en países desarrollados como Suiza, Noruega o el Reino Unido (Robertson, 2013).

En cuanto a países en vía de desarrollo, como India, se encuentran creando estándares y resolviendo temas acerca de la seguridad de estos sistemas (Stone, 2014), en caso de otras regiones como África, el paso de papel a registros electrónicos es mínimo (Candice & Erasmus, 2016). Según Lolimsa, en el 2014, el mismo comportamiento se puede ver en América Latina y en especial en Perú, puesto que un 72% de la información se encuentra en Papel, un 11% se encuentra en un sistema de Historia Clínica Electrónica y un 17% se encuentra en un sistema de medios electrónicos no integrados (El Comercio, 2014).

Esta lenta adopción puede ser debido a diversos factores como los gubernamentales, como se vio anteriormente, así como, a la existencia de a distintas barreras que dificultan la adopción de sistemas de registros clínicos como señalan Ajami y Bagheri-Tadi (2013):

- Tiempo: puesto que el médico no se toma el tiempo para familiarizarse con el producto e inclusive para escoger o averiguar sobre alguno de estos. (p. 131)

- Falta de familiaridad con la computadora: no todos los doctores poseen la habilidad para escuchar los pacientes, tomar notas y discriminar aquello que es médicamente importante. Todo lo anterior, mientras se desplazan por la interfaz

del programa (p.132). Este factor fue demostrado en la implementación de registros médicos en centros de salud de Perú (Revorredo & Cavalcanti, 2014).

- Disrupción en el flujo de trabajo: puesto que ahora los doctores no toman simplemente notas sino requiere de la familiarización con una herramienta para poder tomar notas. (p.132)

- Relación doctor-paciente: la comunicación interpersonal más completa es a través de contacto visual, la cual se pierde mientras que el doctor navega por las interfaces del programa o en busca de botones. (p.132)

- Complejidad: el sistema puede presentar varias pantallas, opciones lo que hacen que sea poco intuitivo y efectivo. (p.132)

Todos estos factores pueden llevar a que este tipo de soluciones sean finalmente dejadas de lado, como demuestra una investigación del 2009 que el 73% de las implementaciones de sistemas de registros electrónicos no se usan en la forma en la que deberían ser usados (Stone, 2014).

Por otro lado, al no adoptar estos, se dejan de lado todos los beneficios que puedan brindar como:

- Beneficio societario: puesto que facilita y da pie a investigaciones, así como, realizar análisis cuantitativos sobre los datos, conocer datos sobre poblaciones e inclusive monitorear o detectar amenazas biológicas. (Menachemi & Collum, 2011)

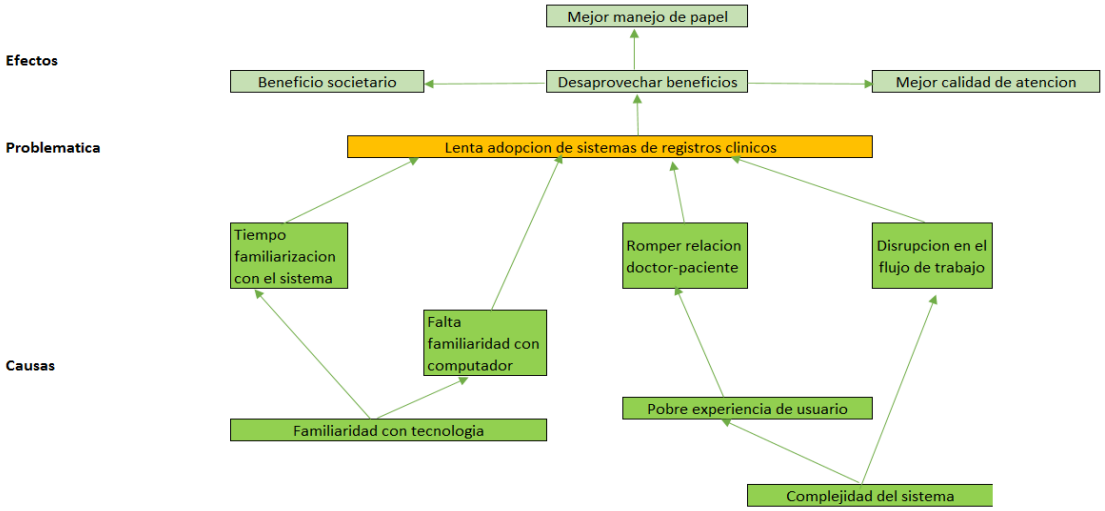
- Menor manejo de papel y menos problemas al gestionar estos, es decir, su organización, mantenimiento en un lugar físico es eliminado puesto que con sistemas de registros electrónicos no requiere de manejo de papel. (University Alliance, 2016)

- Mejor calidad de atención, puesto que se tiene la información de un paciente en un solo lugar por lo que se puede realizar un mejor diagnóstico con menos errores. (University Alliance, 2016)

Otros beneficios que se pueden obtener es la generación de conocimiento y mejor toma de decisiones a partir de los datos obtenidos y el uso de estos para estudios o investigaciones futuras. (Hayri, 2006). Como se puede apreciar en el siguiente diagrama de árbol en la figura 1.1, se resaltan los efectos, causas y la problemática.

Figura 1.1

Diagrama de árbol acerca la lenta adopción de sistemas de registros clínicos



Fuente: Stone (2014)

Luego de haber analizado las barreras que dificultan la adopción de estos sistemas, y teniendo en cuenta que la falta de adopción de estos sistemas puede llevar a dejar pasar por alto distintas oportunidades. Lo anterior, lleva a plantear una solución que no sea intrusiva con el trabajo del doctor, es decir, pueda mantener la interacción con el paciente. Además, que pueda centrarse a este durante la consulta, así como de obtener todos los beneficios que estos sistemas de registros electrónicos pueden brindar.

La solución planteada en el presente trabajo de investigación consiste en la digitalización de los registros clínicos apuntados por el doctor de forma tradicional mediante técnicas de reconocimiento de caracteres óptico (OCR, por sus siglas en inglés). Posteriormente, estructurar estos datos usando técnicas de procesamiento del lenguaje natural, resultando en un proceso no intrusivo al flujo de trabajo del doctor y al mismo tiempo se podrá obtener los beneficios previamente mencionados.

1.2. Objetivo de la investigación

1.2.1. Objetivo general

Desarrollar un sistema que permita estructurar los datos de diagnósticos médicos manuscritos, digitalizándolos mediante el uso técnicas de OCR y procesamiento del lenguaje natural.

1.2.2. Objetivos específicos

- Identificar y aplicar un algoritmo de OCR de buen desempeño para poder realizar la digitalización de los diagnósticos médicos más precisa.

- Obtener las fuentes de datos que se utilizarán de referencia para la categorización, así como los criterios de clasificación.

- Estructurar un algoritmo o técnica de clasificación para el procesamiento del lenguaje natural y así realizar una efectiva categorización.

1.2. Justificación

Día a día gran cantidad de datos acerca de un paciente son generados y recopilados, tomando como ejemplo a una pequeña clínica con cinco consultorios los cuales podrían atender a 3 personas en promedio por hora (Outomuro, 2013). Lo anterior nos lleva a concluir, que durante una jornada de ocho horas muy ocupadas podrían llegar a atender

hasta 120 personas durante el día. En consecuencia, se pudo haber recopilado información acerca de 120 personas, su diagnóstico y las recomendaciones respectivas del doctor.

Estos datos permanecen en un estado sin posibilidad de ser analizados, junto con las desventajas relacionadas a mantener registros clínicos en un formato físico. Lo anterior presenta la oportunidad de digitalizar estos datos al momento que sea generada por el doctor y mantenerla en forma estructurada, para así poder realizar un análisis futuro de esta información y aprovecharla. Por ejemplo, se impacta en las organizaciones orientadas a la salud, puesto que a estas se les está abriendo el paso a realizar un diagnóstico preventivo sobre los datos del paciente (Mathias, et al., 2013), lo cual podría aumentar el tiempo de vida promedio o reducir la tasa de mortalidad (Smith, 2011).

En cuanto la organización, el doctor puede seguir trabajando siguiendo el flujo normal de trabajo manteniendo toda su atención sobre el paciente. En el ámbito político, según la Ley N° 30024 acerca del historial clínico electrónico, pide que los hospitales manejen una sola base de datos acerca de los pacientes. Por lo tanto, con esta solución se podría dejar a la organización con una base para operar con este sistema que el gobierno plantea.

Sobre el ámbito económico de una empresa, se estaría reduciendo el tiempo en administración del papel generado en consulta, por lo que también significa menos problemas de almacenamientos de estos (Smith, 2011). Adicionalmente, significa ahorro en espacio, por lo que un doctor no perdería tanto tiempo en la gestión de estos. Además, significa más horas de atención de pacientes y mayor número de pacientes atendidos (Smith, 2011).

El propósito principal de la presente investigación es desarrollar un sistema que no sea intrusivo al trabajo del doctor. Por lo tanto, este podrá mantener atención completa del paciente sin tener alguna carga física o mental, y así evitar todas las desventajas antes mencionadas de este tipo de sistemas. Para ello, se utilizará un algoritmo de OCR el cual permitirá el reconocimiento de alta precisión de la escritura manual del doctor, esto para poder digitalizar el documento.

Una vez realizado los pasos anteriores y se tenga el documento digitalizado, pasará por técnicas para el procesamiento del lenguaje natural. Las anteriores, deberán permitir el tratamiento de estos datos para que puedan ser aptos para una posterior clasificación y ser guardados en una base de datos de forma estructurada.

1.3. Aportes

En el desarrollo del presente trabajo, durante la búsqueda de información sobre este tipo de sistemas, se encontró que varios de los trabajos iniciaron con registros digitales, o se enfocan en la detección de cierta categoría de palabras a partir de investigaciones en diversos repositorios médicos como MedLine. En otros casos, existían trabajos de investigación donde se inicia desde el papel, sin embargo, no llegaban a la clasificación de toda la información, puesto que el trabajo se centraba en validar técnicas de clasificación.

Por otro lado, en algunas tesis, de licenciatura y doctorales, trataban igualmente el proceso OCR y almacenaban el resultado en una base de datos, sin embargo, la fuente de comparación era distinta a la que se utilizará en el presente trabajo de investigación.

Al presente, en el Perú aún se está en un proceso para llevar al sector de salud al nivel tecnológico. La digitalización de la información, bajo la ley que se decretó aún tiene objeciones u observaciones por el tema de tratamiento de información sensible. Hasta entonces, algunas soluciones han sido crear sistemas con las desventajas antes mencionadas.

Inclusive, no todos los centros médicos poseen los recursos necesarios para su respectiva implementación o los doctores tienen poca voluntad para utilizar estos sistemas pues suelen estar acostumbrados al uso de hoja y papel. Por ello, se resisten al cambio, el cual es un comportamiento natural en este sector, como se aprecia en el artículo de Forbes, donde se analiza por qué la resistencia a la innovación en el sector salud (Smythe, R., 2014).

En este contexto, la innovadora propuesta de solución que se ofrece en el presente trabajo explora técnicas de reconocimiento óptico de caracteres y procesamiento de lenguaje natural, con el fin de digitalizar y categorizar los diagnósticos médicos. Para

ello, utilizará recursos que se pueden encontrar libremente en internet. Además, al estructurar esta información, se encontrarán listos para su futuro análisis y explotación en beneficio del centro de salud.

Esta solución, es una herramienta que permitiría dar el siguiente paso del sector a un plano digital. Adicionalmente, estas herramientas estarán construidas teniendo en cuenta el castellano, un idioma algo ajeno a este tipo de trabajos.

CAPÍTULO II: REVISIÓN DE LITERATURA

En la realización del trabajo de investigación se revisaron diversas fuentes de información, como los artículos publicados por la Sociedad Española para el procesamiento del Lenguaje Natural, National Center for Biotechnology Información, ProQuest, International Association for Management of Technology entre otros. Los trabajos revisados están relacionados a las técnicas de OCR y NLP, así como, fuentes de información para lograr una efectiva categorización.

Con respecto al OCR se revisaron algunos textos que permitieron conocer y familiarizarse con los procesos relacionados a la digitalización de registros clínicos donde esta técnica es aplicada. Sobre los textos revisados, tienen en común que exploran una forma de automatizar algún proceso relacionado al campo de la salud utilizando OCR.

El proceso para la digitalización de texto tiene como primer paso la obtención de datos. Al respecto, los artículos tienen distintas fuentes de donde se obtienen los textos a digitalizar, se pueden encontrar en un formato físico como el papel utilizando un escáner en lote para la digitalización (Biondich, Overhage, Dexter, Downs, Lemmon y McDonald, 2002), a través de un medio digital como una tableta (Bushinak, AbdelGaber y AlSharif, 2011) o de fichas médicas previamente escaneadas y almacenadas en un servidor (Rasmussen, Peissig, McCarty y Starren, 2012).

Las fichas médicas que se trataron en cada artículo revisado presentaban distintos formatos y para enfocarse en solamente lo importante se tuvo que definir regiones de interés de aquello que se quiere procesar. Debido a que Biondich et al. (2002) creó un formato a medida, ya se conocía de antemano los puntos de interés en la hoja (p. 2). A diferencia de Bushinak et al. (2011), sus fichas médicas no seguían un formato específico e inclusive no mencionó un punto de interés específico, sino digitalizaba todo lo que el doctor escribía (p. 8).

Para las siguientes etapas, requiere que se utilice algún tipo de software ya sea de pago o código abierto, e inclusive se puede usar una combinación de los dos como realizó Rasmussen et al. (2012) para reducir los costos de su solución (p. 2.). Por otro lado, Biondich et al. (2002), al utilizar solamente un software de pago tuvo que crear un formulario especial para facilitar el procesamiento del OCR (p. 2.).

Un paso obligatorio para mejorar el rendimiento del OCR es realizar un pre procesamiento de la data, en el caso de haber usado un formulario se tiene que eliminar las líneas o cuadrículas presentes en este. Rasmussen et al. (2012) explica que tuvo que eliminar el ruido introducido por el escáner e inclusive realizó algoritmos específicos para eliminar las líneas de los formularios (p. 3.). Este paso puede ser casi obviado si registras la escritura sobre una tableta como Bushinak et al. (2011).

El paso siguiente que se observa es el de procesamiento, es decir, aplicar algún motor o algoritmo que permita realizar el OCR. En este paso, Rasmussen et al. (2012) demostró que es posible aplicar múltiples algoritmos de OCR para realizar el reconocimiento, tanto de forma individual como combinados (p. 3.), a diferencia de otras investigaciones que simplemente utilizaron las capacidades que su software de pago permitía.

Sobre lo anterior, Rasmusen et al. (2012) antes que se pueda realizar una clasificación de caracteres tuvieron que realizar entrenamiento de sus datos sobre Tesseract y sobre sus otros 2 motores comerciales no, puesto que venían previamente entrenados (p. 3.). Sin embargo, el algoritmo de Tesseract fue entrenado solamente con aquellos caracteres específicos a su investigación a diferencia de los otros dos motores que contenían todo el alfabeto inglés, número y caracteres de puntuación.

Por otro lado, Biondich et al. (2002) realizaron el procesamiento dos veces para asegurar la consistencia del software (p. 3.). Además, mostraron una forma única de evaluar el procesamiento para cada letra de forma individual, puesto que a cada formato digitalizado se le asignó un umbral superior e inferior de confianza, así mismo, cuando el software analiza el dígito se le asigna un nivel de confianza el cual indica la certeza del software para identificar el dígito.

Con lo anterior, se establecieron las siguientes reglas, se acepta como correctamente procesado si el nivel de confianza es mayor al umbral, si está entre ambos valores, se clasifica el dígito pero es marcado, si está debajo del umbral entonces solamente es marcado. Cuando un carácter es marcado, significa que una persona posteriormente pueda corregir o revisar dicho dígito de forma casi manual puesto que se muestra la parte exacta del formato donde está el dígito y la clasificación del sistema, en caso de existir.

Como siguiente paso se tiene el de postprocesamiento, el cual es utilizado para mejorar los resultados obtenidos en el paso previo en base a las peculiaridades de los datos procesados. Según Bushinak et al. (2011), el texto puede presentar distintos errores ya sea por culpa de la herramienta o del propio doctor al ingresar el texto, durante la escritura o por algún otro factor (p. 5), esto hace que el presente punto sea importante.

La propuesta de Bushinak et al. (2011) para este paso es realizar una corrección ortográfica donde el programa propondrá al médico correcciones de las palabras que detecta como incorrectas (p. 5). Para ello, utilizará como base un diccionario médico el cual contiene adjetivos, verbos, pronombres, entre otros, relacionados al campo de la medicina.

Por otro lado, Rasmussen et al. (2011) establecieron ciertas reglas a partir de patrones que se pudieron obtener de resultados no tan exitosos, estas reglas las llamaron optimización sensible al contexto (p.3), puesto que los motores lograron obtener resultados incorrectos para ciertos caracteres que se encontraban juntos. Razón por la cual se definieron ciertas reglas con las que se pudo remediar en cierto grado la solución.

Un ejemplo de esta regla es la siguiente, se encontró que uno de los caracteres problemáticos fue la “r” minúscula, dado que puede ser reconocido como uno de los siguientes caracteres “1”, “I”, “L”, “/” o “\”, inclusive esos caracteres previos pueden ser confundidos entre sí. Además, el símbolo “+” puede ser confundido con la letra “t” y del mismo modo “Z” con “2”.

Finalmente, el último paso que realizan es la de validaciones sobre la solución. Biondich et al. (2002), realizaron mediciones de tanto del nivel de confianza y exactitud de los niveles de confianza (p. 3). Debido a que en su investigación intervenían tanto

enfermeras como doctores pudo comparar si había alguna diferencia en la exactitud del clasificador, sobre esto no encontró mayor diferencia. Por otro lado, también pudo evaluar la usabilidad y velocidad de este sistema comparado al proceso netamente manual realizado por los digitadores de su clínica, encontrando que su solución era más rápida que ellos.

En cuanto a Rasmussen et al. (2011), realizaron mediciones utilizando los valores de precisión, valor negativamente predecible, sensibilidad, así como, especificidad para cada motor de OCR y combinación de estos (p. 3). Con ello, pudieron encontrar que las reglas establecidas en el procesamiento no tuvo una mejora significativa pero la combinación de múltiples motores de OCR tuvo un resultado positivo.

Un punto de discusión común entre algunos autores es que, debido a las diversas variaciones en la escritura de las personas, siempre sería necesario una revisión manual de los resultados, pero estos pueden ser mejorados si el personal recibe entrenamiento. Adicionalmente, si el presupuesto para una solución similar es flexible, se podría utilizar alguna herramienta de OCR de alta precisión.

La investigación de Bushinak et al. (2011) resalta porque muestra en alto nivel como es el flujo de los datos y la integración de distintas herramientas con el objetivo de estructurar la información. Sin embargo, durante sus pruebas el doctor se encuentra limitado a escribir en una tableta con un lapicero digital, además de usar un OCR comercial.

Una vez culminada la digitalización de los diagnósticos se debe iniciar el procesamiento al texto para poder categorizarlo eficazmente. La primera definición para categorizar las enfermedades es establecer las fuentes de datos que proveerán dicho diccionario entre término y categoría.

El artículo “Detección de fármacos genéricos en textos biomédicos” presenta la compleja tarea de categorización de fármacos. Esta investigación presenta dos fuentes de información: UMLS y reglas de USAN. El primero es el Sistema de Lenguaje Médico Unificado (UMLS), que se define como una base de datos de conocimiento que integra varios recursos con el fin de facilitar el desarrollo de sistemas automáticos para el procesamiento del lenguaje natural en el dominio de la biomedicina (Segura I., 2008).

La segunda herramienta son las reglas de nombrado recomendadas por el consejo U.S. Adopted Name (USAN), que es la institución responsable de la creación y asignación de un nombre genérico a un nuevo fármaco, están especificadas como prácticas actuales para nombrar fármacos y recaen en el uso de afijos (Segura I., 2008).

Con la misma fuente, pero con otra metodología el artículo "Categorización de textos biomédicos usando UMLS", utiliza el metatesauro UMLS que incluye varias ontologías médicas (Perea, J. et al., 2009).

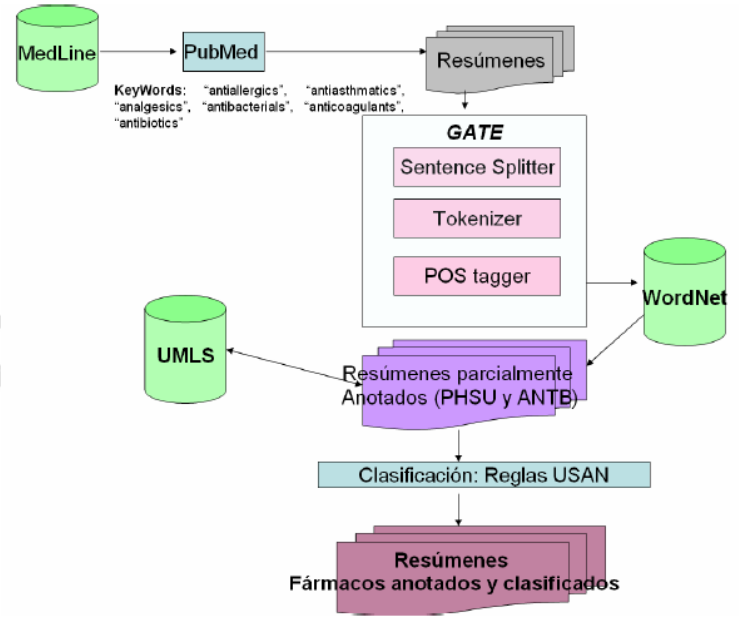
Por último, la investigación titulada como "The aid of machine learning to overcome the classification of real health discharge reports written in Spanish", evidencia el problema con el lenguaje utilizado por los doctores y las jergas u otros que presentan obstáculos en el análisis como los acrónimos, omisiones, sinónimos, errores al escribir u omisiones. Este artículo utiliza información del Hospital Galdakao-Usansolo en España, y presentan el trabajo como la clasificación asociada al ICD-9. Califican el trabajo como uno de minería de texto enfocándose en el emparejamiento del término con el código ICD-9 y como principal objetivo desarrollar un sistema de clasificación computacional con la mayor precisión posible.

La definición del diccionario que servirá para la categorización es sumamente relevante, como se expuso en dos artículos UMLS contiene gran cantidad de información, mientras que otra opción también es ICD-9 que se puede subir simplemente a una base de datos, a diferencia de UMLS que se tiene que integrar a la solución. Con esta definición de fuentes, el siguiente proceso es el tratamiento del texto del diagnóstico para lograr una eficaz clasificación, excluyendo todo lo que no sea relevante para el análisis.

El artículo "Detección de fármacos genéricos en textos biomédicos", describe la arquitectura del sistema en tres módulos: En el primer módulo, inicia el tratamiento del texto dividiéndolo en oraciones, se identifican los tokens y se analiza morfosintácticamente. En el segundo módulo se busca en el Metatesauro de UMLS cada término que no estaba en WordNet. Por último, se implementa las recomendaciones del consejo USAN, con lo que se devuelve la lista de los afijos que están contenidos consiguiendo así posibles familias farmacológicas (Segura I., 2008).

Figura 2.1

Arquitectura de la solución de los tres módulos.



Fuente: Segura L. et al (2008)

La figura 2.1 presenta la arquitectura del sistema donde interactúan todos los componentes y asimismo los tres grandes procesos en la metodología, como se explicó anteriormente en los tres módulos respectivos.

Asimismo, el artículo "Categorización de textos biomédicos usando UMLS", sigue una metodología similar, a diferencia de la falta de filtros como las reglas USAN. Mientras, el trabajo "The aid of machine learning to overcome the classification of real health discharge reports written in Spanish" no ofrece detalles precisamente del tratamiento de información, sino se enfoca más en las técnicas de machine learning utilizadas.

A diferencia de los dos últimos artículos, el primero expande su explicación en la metodología que utiliza para el tratamiento de la información. El presente trabajo ha tomado conocimiento de la metodología para guiar el tratamiento que se dará a la información. En última instancia, luego de darle el tratamiento adecuado al texto se procederá a la clasificación y su respectiva validación de resultados.

En el artículo “Detección de fármacos genéricos en textos biomédicos” son procesados 1481 resúmenes, utilizando solo UMLS se llegó a realizar una identificación del 97% de los términos con un 100% de precisión en la categorización. En suma, con el filtro adicional de las reglas USAN logró detectar un total de 99.8% de cobertura, aunque con un 99.3% de precisión, como se observa en la siguiente tabla.

Tabla 2.1

Resultados de cobertura y precisión del experimento

	Cobertura	*Precisión*
UMLS	97%	100%
UMLS + Rules	99.8%	99.3%

Fuente: Segura L. et al, (2008).

El tener menor cobertura al encontrar los términos solo con la base UMLS, se debe a que esta no se actualiza diariamente, mientras las reglas si son actualizadas constantemente. Por otro lado, la precisión al encontrar los términos se disminuye con las reglas de la USAN, pues había errores en la clasificación por la terminología.

Por otro lado, el artículo “Categorización de textos biomédicos usando UMLS” las técnicas usadas fueron Support Vector Machine (SVM) y una red neuronal tipo perceptrón denominada PLAUM. Para estos no se ha considerado ninguna variación de parámetros y los resultados demuestran que la técnica SVM es mejor que PLAUM cuando no se aplica expansión de términos, mientras que cuando se aplica, PLAUM resulta tener mayor éxito.

Mientras, el trabajo “The aid of machine learning to overcome the classification of real health discharge reports written in Spanish”, las técnicas que se experimentaron fueron diferentes modelos de aprendizaje de máquina: árboles de decisión, naive bayes, regresión logística y SVM usando datos estructurados.

Se propuso Naive Bayes como línea de base porque ha sido probado en otras tareas de minería de texto, sin embargo, para el escenario de la presente investigación tuvo pobres resultados, teniendo mejores resultados el algoritmo de Random Forest en los datos de entrenamiento y SVM en los datos evaluados.

El aporte de los artículos de investigación expuestos sobre clasificación de información biomédica en conjunto se tomarán en cuenta para la realización del presente trabajo de investigación, como por ejemplo la metodología del tratamiento de los diagnósticos, las técnicas utilizadas para el procesamiento del lenguaje natural y sus respectivas métricas en la validación.

CAPÍTULO III: MARCO TEÓRICO

En el siguiente capítulo, luego de haber revisado la literatura investigada, se presenta a detalle las distintas metodologías, técnicas y conocimientos específicos que respaldan las bases del presente trabajo de investigación.

3.1. Healthcare Information Technology (HIT)

La solución basada en la presente investigación sería considerada como una Health Information Technology (HIT). Como mencionan en el libro "Introducción al HIT" (Ciampa & Revels, 2013), HIT ha sido descrito de diversas maneras como un framework o mecanismo que permite la gestión de información referente a la salud o destinada a un mejor cuidado del paciente. Finalmente, todas hacen referencias a la aplicación de las tecnologías de información en la industria de la salud.

HIT se puede definir también, en un concepto un poco más técnico: es el uso de hardware y software en el esfuerzo para gestionar y manipular datos e información de salud (Ciampa & Revels, 2013). Inclusive solamente con esta definición, se deja de lado otros aspectos que puede considerar HIT; como la seguridad, sobre la cual han llegado a desarrollar distintas regulaciones por partes de gobiernos, como es el caso de Estados Unidos, incluyendo en su legislación a Health Insurance Portability and Accountability Act (HIPPA) o Health Information Technology for Economic and Clinical Health (HITECH) (Wilson & McEvoy, 2011). En otros casos, investigadores han llegado a desarrollar marcos de trabajo para lograr este fin (Singh & Sitting, 2015).

Uno de sus más importantes usos es el de reducir errores médicos, debido al impacto que puede tener HIT sobre la calidad y seguridad médica al basarse en sistemas de decisiones automatizados y herramientas de adquisición de conocimientos. Otro beneficio es sobre la atención ambulatoria, donde puede otorgar mejoras de eficiencia sobre el uso de sistemas que permitan el manejo de citas y administración de finanzas, todo actuando sobre procesos conjuntamente con su mejora.

De forma similar, el uso de Electronic Health Records (EHR) permite mayor facilidad a la hora de acceder a información y reducción de uso de papel o documentación física. Adicionalmente, se podría lograr la integración con otras organizaciones, proveedores por medio de intercambio de información regional o inclusive global (Shekelle, Morton, & Keeler, 2006).

Este concepto también es referido como un paragua de términos que describen iniciativas basadas en computación en la salud (International Resources Management Association, 2013), algunos términos que son usados con frecuencias dentro de este campo son:

- Electronic Health Record (EHR), este representa el componente central en una infraestructura de HIT (Rouse, 2015), el cual surge como alternativa a la forma tradicional de tomar apuntes sobre un paciente en un formato físico que es el papel, lo cual tenía como desventaja que no permitía que un doctor este enterado de todo el historial pasado de la persona. Entonces, con un EHR, se puede tener acceso a mayor cantidad de información sobre un paciente y población. Además, indica en forma genérica a cualquier sistema que permita el cuidado de pacientes en forma electrónica, habilitando la capacidad de poder intercambiar la información que estos mantienen con otras organizaciones de salud.

- Patient Health Record (PHR), es otro de los elementos esenciales del HIT (Rouse, 2015), el cual consiste en que el historial es manejado y controlado por el paciente. Este puede contener información acerca de alergias, enfermedades, resultados de laboratorio, entre otros. Además, podrá tener la capacidad de conectarse al EHR del doctor (HealthIT, 2016).

De acuerdo a Peter Waegamann (Wargemann, 2003), existen otros términos similares a los anteriores pero que también son usados con frecuencia en la literatura, como son:

- Clinical Patient Record (CPR), hace referencia a todo el historial a lo largo de la vida de un paciente tomando en cuenta todas las especialidades que lo atendieron.

Además, de poseer una capacidad de comunicarse con diferentes sistemas a nivel internacional.

- Electronic Patient Record (EPR), es un concepto similar al CPR el cual se centra en información relevante, por lo que no considera la historia dental, comportamiento o de tratamientos alternativos.

- Electronic Medical Record (EMR), hace referencia a un registro electrónico del paciente con capacidad de comunicarse con distintos sistemas dentro de una organización.

Muchos de estos términos involucran tener la capacidad de interoperabilidad entre distintos sistemas u organizaciones del sector salud, donde puedan intercambiar información y usarla, todo esto es de forma transparente. Por lo expuesto anteriormente, la Office of the National Coordinator for Health Information Technology (ONC) está trabajando en crear estándares que permitan interoperabilidad.

La capacidad de interoperabilidad es diferente al Health Information Exchange (HIE), este implica que la información pueda desplazarse entre diferentes organizaciones, es decir, se puede acceder a la información proveniente de múltiples organizaciones, su gran diferencial con interoperabilidad es que no pretende que esta pueda ser usada (HealthIT, 2016).

3.2. Visión de computadora

Es aquella que permite el análisis de imágenes usando computadoras para poder realizar una descripción de aquellos objetos que sean captados por la cámara, es decir, que un sistema que aplique este concepto presentará como entrada una imagen y como de salida es información (UC3M, 2016). Este último punto representa la mayor diferencia que presenta con respecto a otras técnicas, como la del procesamiento de imágenes digitales, la cual se enfoca en la modificación de la imagen para mejorar distintos aspectos de esta

para que luego pueda ser utilizado o aprovechado por una persona (Sucar & Gomez, 2011).

Algunas ventajas que se pueden obtener de la visión por computadora son en el campo de aplicaciones industriales, en la cual las maquinas o robots pueden realizar controles de calidad, clasificación de componentes, puede ser usado en una variedad de campos como biomedicina, militares, robótica, agricultura, seguridad, entre muchos otros, como la de campos emergentes (De la Escalera & Armingol, 2010).

El contexto en el que se encuentra este campo es que el mundo actualmente captura bastante información puesto que las cámaras están bastante inculcadas en la sociedad actual. Por otro lado, los procesadores son cada vez más veloces y se reducen en tamaño, por lo que ahora es posible manejar grandes cantidades de información en menor tiempo e inclusive realizar estas operaciones en una gran variedad de dispositivos (UC3M, 2016). Todo esto es movido por distintos algoritmos que son de conocimiento público, por lo que investigaciones pueden partir de ese punto y los investigadores pueden desarrollar su idea en base a la posible aplicación de distintas bibliotecas (UC3M, 2016).

Una de las bibliotecas a las que se puede hacer referencia es OpenCV, la cual posee todo un conjunto de funciones para aplicaciones de visión por computador. Esta biblioteca ayuda al avance de esta área otorgando código de acceso libre y cuyo uso ha sido mejorado y es específico para poder implementar algoritmos que permitan el análisis de imágenes (UC3M, 2016).

La historia de OpenCV comienza por iniciativa de Intel en el año 1999, la cual desarrollo en fase beta hasta el año 2005 y ya para el siguiente año se publicó su primera versión en C, la cual fue siguiendo usando en su segunda versión, posteriormente el uso de esta librería fue extendido para poder ser usado por el lenguaje C++. Esto marcó un punto en el cual se fueron implementando una gran variedad de funciones llegando a la actualidad a más de 500 funciones, sin contar los algoritmos productos de investigaciones que se implementan con estas librerías fue OpenCV (UC3M, 2016).

3.3. Optical Character Recognition (OCR)

Reconocimiento óptico de caracteres, en su abreviación en inglés OCR, es aquel proceso que consiste en transformar el texto en papel, sea impreso o escrito a mano, en un formato que pueda ser entendido y manipulado por la computadora (Ahmed, 2005). Como se menciona en "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey" (Plamondon, 2000), distintas técnicas pueden ser asociadas con una escritura hecho a mano por lo que señala las siguientes:

- Reconocimiento de escritura manual, el cual consiste en convertir representaciones lingüísticas a su representación simbólica típicamente en su representación ASCII.

- Interpretación de escritura manual, reconocer que significa una determinada parte de un documento.

- Identificación de escritura manual, consiste en reconocer a quien le pertenece un determinado escrito, teniendo en cuenta que la forma de escribir de una persona es única.

- Verificación de firmas, permite validar si una determinada firma le pertenece a una persona.

Según Plamondon (2000), existen dos métodos para realizar las tareas de digitalización de textos que serían las de reconocimiento offline y online. En el caso de reconocimiento online, consiste que mientras la persona escribe, distintas coordenadas de puntos sucesivos son guardados, por lo que se podría reconocer los caracteres mientras se escriben (Eikvil, 1993).

En el caso del reconocimiento offline, la escritura entera está disponible, es decir, se encuentra en una representación visual a partir de la cual se puede aplicar un análisis, por lo que esta ocurre después que la escritura ya tomó lugar. Como señala en "Optical Character Recognition" (Eikvil, 1993), mientras más se pueda mantener el control sobre

el contexto en el que se encuentran estas palabras el análisis de esta será más fácil. En otras palabras, si estas se mantienen en cajas separadas, mayor será el rendimiento del OCR, por otro lado, cuando el texto no es controlado, como es el caso de escritura manual representan un reto en esta área.

Por otro lado, de estos dos métodos, se ha demostrado que se pueden obtener mejores resultados con el reconocimiento online. Lo anterior es debido a que se pueden recoger variables como posición, velocidad o aceleración en función del tiempo en que se escribe por medio de una pantalla o superficie táctil, y con ayuda de algún lapicero especial.

Además, el reconocimiento online se puede aprovechar en ciertos campos; como es la computación basada en lapicero, que trata de imitar la funcionalidad de un lápiz o lapicero de forma electrónica; verificadora de firmas, explicado anteriormente. Además de otras aplicaciones, como procesamiento automático de escritura manual, ayudar a niños a aprender a escribir o ayudar a personas con discapacidades para recuperar funciones motoras, entre muchas otras (Plamondon, 2000).

En el caso de reconocimiento offline, se sigue una estructura similar para distintos tipos de soluciones, donde una primera etapa es la de adquisición de la imagen, que consiste en obtener el texto escaneado o con una foto en algún tipo de formato digital como puede ser JPEG, BMP, etc. (Vashisht & Nandal, 2016). Algunas aplicaciones que tiene el reconocimiento offline son para interpretar direcciones escritas para la entrega de correos físicos; reconocimiento de cheques bancarios, los cuales tienen palabras en distintos formatos e inclusive dígitos y firmas, además, de otros métodos son los de verificación de firmas e identificación de escrituras (Plamondon, 2000).

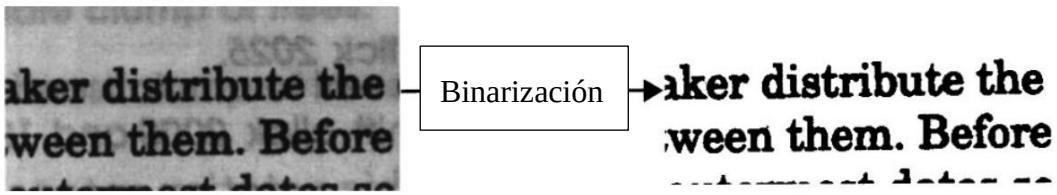
A parte del reconocimiento de caracteres para el cual es necesario ejecutar pasos previos, a este conjunto de pasos se le conoce como la de pre procesamiento. Es en esta etapa donde se realizan ciertas técnicas al texto previamente obtenido en la etapa anterior con el fin de incrementar la precisión del algoritmo de reconocimiento. Algunos pasos consisten en la eliminación de todo tipo de ruido que la imagen pueda presentar, así como, eliminar aquellos patrones que hacen única la escritura de persona a persona (Vashisht & Nandal, 2016), como los que explicaremos a continuación.

Uno de los pasos que se realiza se conoce como binarización, como su nombre lo indica permitirá convertir una imagen de pixeles en una imagen binaria siguiendo un proceso, la cual tiene como primer paso convertir la imagen obtenida a una escala de grises (Niklas, 2016). Si se obtiene un histograma de su escala de grises esta imagen presentará dos picos, un pico alto y uno pequeño, el primero es asignado a un fondo blanco y el segundo es asignado al color negro.

A partir de los picos mencionados anteriormente, se tiene que determinar un valor optimo que será el umbral el cual si es superior a este equivale a blanco y debajo de este equivale a negro, este paso es conocido como thresholding (Plamondon, 2000). El umbral puede ser de valor fijo (para todo el documento) o adaptativo (por cada pixel se usan diferentes valores dependiendo de una determinada cantidad de pixeles próximos a estos) (Vamvakas, 2014), un algoritmo utilizado para es Iso Data Algorithm (Niklas, 2016). Esta parte se realiza con el fin de separar el texto del fondo que lo contiene, además, suavizará la imagen (Plamondon, 2000), como se ve en la figura 3.1.

Figura 3.1

Ejemplo de binarización sobre un texto.



Fuente: Sauvola & PietikaKinen, (1998).

Entre los pasos siguientes consisten en la eliminación o reducción de ruidos, el cual se tratará de obtener los textos quitando todas aquellas manchas producto de interferencias que se pudo obtener a partir de su escaneo, transmisión, entre otros, para lo que se suele emplear técnicas de suavizamiento (Plamondon, 2000), como se puede ver en la figura 3.2. Uno de los retos para la escritura manual es que se tiene que mantener las conexiones entre las palabras (Plamondon, 2000), para lo que existen otro conjunto de algoritmos.

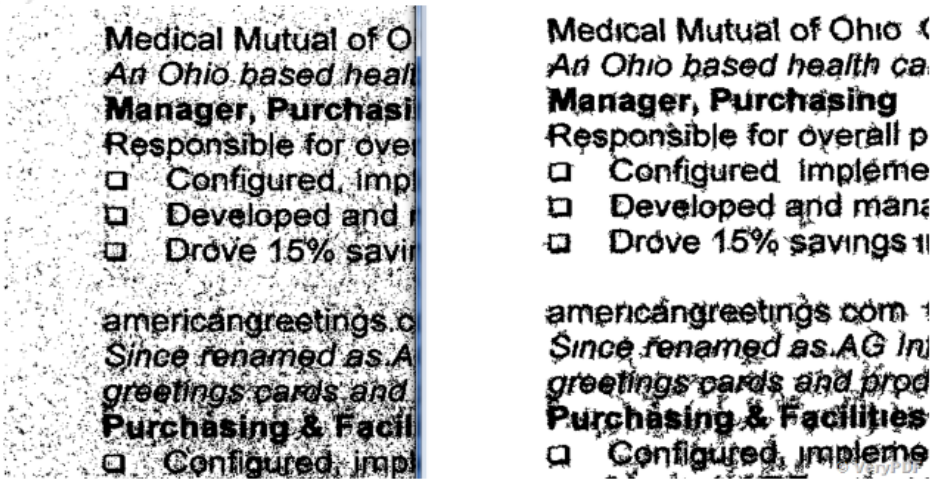
Existen distintos tipos de ruido que puede presentar una imagen, entre los tipos de ruido principales que se pueden encontrar, como es el ruido gaussiano, en el cual la

imagen presenta pequeñas variaciones (UC3M, 2016). Otro tipo de ruido es el impulsional, es el más común y conocido como sal y pimienta, estos son puntos que están dispersos sobre toda la imagen como si efectivamente sal y pimienta hubiese sido rociada sobre esta (Vashisht & Nandal, 2016), esto ocurre porque la imagen toma valores mínimos o máximos (UC3M, 2016). Algunas técnicas de reducción de ruido son:

- Filtrado, se diseñan filtros para eliminar el ruido en una imagen que pueden haber sido ocasionados por una superficie desnivelada, como puntos que aparecen de forma aleatoria en una imagen. (Vashisht & Nandal, 2016).

Figura 3.2

Ejemplo de reducción de ruido sobre un texto.



Fuente: VeryPDF (2015).

- Operaciones morfológicas, estas vienen a ser las diferentes técnicas que sirven para poder obtener la forma de una determinada región y modificarla, con ello es posible remover el ruido de un documento ocasionado por la tinta y la calidad del papel (Vashisht & Nandal, 2016). Algunas de las operaciones son: erosión, se tiene un elemento estructural que recorre los alrededores del objeto con el fin de degradar la imagen; dilatación, a partir de un elemento estructural, se bordeará el objeto haciendo de este más grueso o en otras palabras se expandirá; apertura, consiste en aplicar una función de erosión y posteriormente una de dilatación; cerrado, consiste en aplicar una función de dilatación y posteriormente una de erosión (De la Escalera, Visión por Computador. Fundamentos y métodos, 2011),

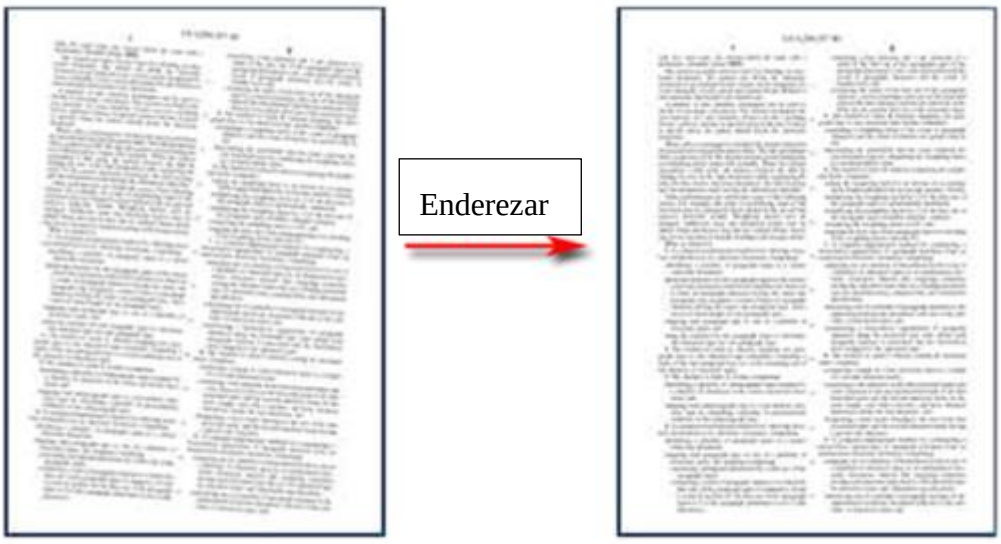
el cual si es aplicado correctamente permitirá eliminar segmentos no unidos en la imagen o agujeros en la imagen (The University of Utah, 2012).

- Normalización, esta técnica permite reducir todas las variaciones que pueden ser introducidas por la persona al momento que escribe. Entre algunas operaciones comunes de normalización, se tiene la de redimensionar las imágenes manteniendo el ratio o no (Hu, Brown, & Turin, 2000), incluso corrección de desviaciones como se verá posteriormente.

- Corrección de desviaciones, el objetivo es alinear el texto del documento en un plano horizontal corrigiendo aquella desviación introducida por el escritor, esto se puede ver en la figura 3.3. Algunas técnicas usadas son de correlación, proyección de perfiles, transformada de Hough, entre otras (Vamvakas, 2014).

Figura 3.3 pués de aplicar corrección de desviacion

Antes y después de aplicar corrección de desviaciones o enderezamiento de un texto,



Fuente: VeryPDF (2015).

Otro de los grandes pasos es donde se realiza la segmentación, uno de los tipos de segmentación es el de línea, el cual consiste en dividir el texto entero en distintas líneas

(Plamondon, 2000). En cuanto a un texto impreso, esta tarea se logra al medir las desviaciones o inclinaciones de los ángulos de la línea de un párrafo. Sin embargo, cuando se trata de textos escritos a mano es más difícil, puesto que las palabras pueden tener variaciones y no necesariamente seguir una línea recta (Vashisht & Nandal, 2016).

Para este caso se puede utilizar clusterización para establecer un punto mínimo, donde se puedan establecer líneas, esto se refiere a encontrar y definir la línea imaginaria que las personas establecen cuando escriben a mano (Plamondon, 2000).

Es necesario notar que, en la literatura hacen referencia a la segmentación externa, la cual es una técnica que permitirá dividir la página en distintos elementos como párrafos, oraciones o palabras. Además, permite reconocer regiones con texto o sin este (Vashisht & Nandal, 2016).

Así mismo, existe la segmentación en palabras y caracteres. Esta se realiza luego de haber definido la segmentación de líneas, y se centra en encontrar espacios entre palabras y letras. Este tipo de segmentación asume que los espacios más grandes separan a los primeros y espacios más cortos a los segundos, pero esta regla puede tener excepciones como distintos tipos de trazas, por lo que se propone un método que toma en cuenta las variaciones entre caracteres adyacentes entre sí de un escritor (Plamondon, 2000).

Por otro lado, la segmentación de caracteres puede referirse también a la segmentación interna, la cual a partir de la imagen principal se obtienen subimágenes con un caracter cada uno (Vashisht & Nandal, 2016).

Un paso previo al reconocimiento, pero después del preprocesamiento, es la de extracción de características. Como su nombre lo indica, permite obtener aquellas características necesarias y más relevantes para que posteriormente un clasificador u otro algoritmo lo pueda aprovechar, en lugar de tomar todas las características que pueda presentar un elemento (Storcheus, 2015).

En este caso si se aplica para reconocimiento de caracteres, cada caracter es relacionado con un vector que contiene sus características, por lo tanto, esta fase permitirá aumentar la tasa de reconocimiento (Vashisht & Nandal, 2016). Ejemplo de este seria la técnica de adelgazamiento (Poovizhi, 2014), la cual permite que cada letra tenga un solo

pixel de grosor como se puede ver en la figura 3.4, reduciendo el tamaño de los datos en el proceso.

Figura 3.4

Antes y después de aplicar adelgazamiento en una letra.

Fuente: Arora, Malik, Bhattacharjee, & Nasipuri (2010)

El siguiente gran paso es el de reconocimiento de caracteres, el cual es referido como OCR si son caracteres impresos, en el caso de ser escritos a mano con letra imprenta y que sigue alguna estructura como puede ser una letra por cada cuadricula en un formato (Bishop, 2015) es conocido como Intelligent Character Recognition (ICR).

Un método bastante común es aquel basado en redes neuronales y algoritmos basados en nearest-neighbor, los cuales tienen mayor precisión, pero tienden a ser más lentos (Plamondon, 2000).

La literatura indica que reconocer caracteres de un documento impreso puede ser más preciso e inclusive algunos tipos de letra suelen ser más precisos que otros (Smith, 2016). Por otro lado, reconocer caracteres escritos a mano suelen presentar mayor dificultad, pues requerirá modelos de segmentación a nivel de palabras y letras debido a la naturaleza variable de la letra humana. Por lo expuesto anteriormente, es que se denomina a este campo Handwritten Character Recognition.

Las técnicas y muchas más que se usan en esta etapa descrita pueden ser agrupadas en las siguientes categorías:

- Template Matching, implica utilizar una plantilla la cual representa un caracter que ya se encuentra en una imagen y con ella, encontrar aquella región en dicha imagen con la mayor similitud, para ello toma en cuenta el tamaño de la imagen (Vashisht & Nandal, 2016).

- Métodos estadísticos:
  - Reconocimiento no paramétrico, el cual no requiere de información previa de los datos y que por medio de un clúster, un patrón puede ser clasificado. El método más usado es el de nearest-neighbor o vecinos cercanos (Vashisht & Nandal, 2016), profundizaremos sobre este punto más adelante.

  - Reconocimiento paramétrico, en el cual se obtiene información previa de los datos que serán usados en la etapa de entrenamiento, la clasificación en este tipo de reconocimiento puede utilizar métodos como el de Bayes y máxima similitud (Vashisht & Nandal, 2016).

- Otras técnicas, existen también otras técnicas como la de técnicas estructurales, donde su mayor exponente son las redes neuronales, que poseen una estructura que puede trabajar en paralelo haciendo de esta una técnica rápida a comparación de otras (Vashisht & Nandal, 2016).

Finalmente, se realizará el reconocimiento de palabras, donde un algoritmo asocia la palabra que ya fue reconocida a un diccionario predefinido o conocido también como léxico, es decir, el vocabulario usado en una rama del conocimiento (Merriam-Webster, 2016). Para ello tendrá dos enfoques, un enfoque analítico y el otro holístico, el primero consiste en reconocer caracteres individuales, mientras que el segundo se enfoca en agrupar un conjunto de caracteres y este resulta más útil en escenarios donde los caracteres se tocan entre sí, ya sea que se encuentren impresos o escritos a mano.

El proceso que se puede usar para distintas aplicaciones de OCR mantiene un mismo proceso similar y esencial, como el preprocesamiento, una posible segmentación de palabras, reconocimiento y postprocesamiento. Sin embargo, la segmentación puede ser evitada o lograda de otra forma con un método que permite reconocer la zona donde se encuentran las palabras al "dibujar" el contorno de la imagen de la palabra, prácticamente dilatándola (Plamondon, 2000).

Es necesario mencionar que los métodos para extracción de características son clave para lograr un alto rendimiento en el reconocimiento, un enfoque se basa en la teoría

que la escritura a mano posee características regulares las cuales son modificadas, no siempre, por características singulares de una forma casi "decorativa" (Plamondon, 2000).

3.4. Inteligencia Artificial

El presente trabajo de investigación toma conceptos y definiciones de inteligencia artificial, por lo que es relevante partir de este punto y definir su significado, para así seguir explicando conceptos más específicos. La interrogante planteada por Alan Turing hace medio siglo, rodeando los cincuentas, acerca si las máquinas podrían pensar, fue uno de los primeros cuestionamientos, aunque algo absurdos para su época, que impulsaron la investigación y nacimiento de la inteligencia artificial (Turing, A., 1950).

La inteligencia artificial es definida como un área de la ciencia que crea sistemas capaces de efectuar actividades que requieren de inteligencia para así resolver diversos problemas como lo haría la inteligencia humana (Copeland, J., 2000). Alan Turing intentó demostrar ello en una prueba que retaba la habilidad de la máquina a mostrar un comportamiento inteligente similar al de un humano, esta prueba fue llamada "Turing Test" (Turing, A., 1950).

Uno de los científicos más famosos del mundo moderno, Stephen Hawking, a fines del 2014 ofreció una entrevista a la cadena British Broadcasting Corporation (BBC) donde expresó que el desarrollo completo de la inteligencia artificial podría ser el fin de la raza humana. (Cellan-Jones, R., 2014). Pudiendo ser este un presagio, lo cierto es que en la actualidad las aplicaciones de la inteligencia artificial se pueden ver en diferentes campos, un ejemplo representativo de ello es Siri, la aplicación de Apple que permite preguntarle al dispositivo y darle órdenes mediante el reconocimiento de voz.

Entre la lista de problemáticas donde las técnicas de inteligencia artificial son típicamente usadas, están: Reconocimiento óptico de caracteres, robótica, realidad virtual, procesamiento de imágenes, procesamiento de lenguaje natural, etc. (Nordlandle, E., 2001)

En este contexto, dos de las grandes problemáticas tratadas en el presente trabajo de investigación, son aplicaciones donde típicamente se usan técnicas de inteligencia

artificial. A continuación, se continuará profundizando en los conceptos más específicos en los que se basa el presente trabajo.

3.5. Natural Language Processing

Natural Language Processing (NLP) o Procesamiento de Lenguaje Natural es un campo de la inteligencia artificial cuyo objeto de estudio es la interacción que existe entre el lenguaje humano y las computadoras.

Jaime Carbonell, científico de computación, en una entrevista en el congreso de Sevilla de 1992 lo definió como una disciplina en tecnología en transición y describe como uno de sus principales objetivos el procesamiento de textos (Carbonell, J., 1992).

Según lo estimado en congresos de la IFIP (International Federation for Information Processing) existen datos almacenados en forma de texto tan abundantes que, en otro tipo de estructura hasta la actualidad, sin embargo, esto ya está cambiando pues nuevas estructuras están creciendo como imágenes, videos, audios, entre otros que corresponden a la revolución de big data.

Carbonell también afirma que las necesidades de los usuarios van más allá de la recuperación de información en estos textos, sino también está la extracción de los datos significativos.

Carbonell, por último, señala que la dificultad más difícil del procesamiento del lenguaje natural es la ambigüedad del mismo. La resolución de ambigüedades debe ser bien tratada en un procesamiento eficaz. En efecto, para la presente investigación existen dos problemas en este tipo de ambigüedad, el primero es la letra que tienen la mayoría de doctores y el segundo son los diferentes sinónimos que manejan los colegas.

Respecto al primero existen investigaciones ya realizadas (Lyons, R. et al, 1998), como se observa en la tabla, que señalan que en la identificación de letra de doctor es aún difícil para los mismos pacientes y como se puede ver en la tabla hasta las enfermeras y administrativos tienen errores debido a la legibilidad de la letra del doctor.

Tabla 3.1

Legibilidad en la letra de doctor

Median legibility error score of each occupational group

	Median error score (interquartile range)		
	Doctors	Nurses and other medical professions	Administrative staff
All subjects:	(n=38)	(n=32)	(n=22)
Letters of alphabet*	7 (0-10)	3 (1-6)	4 (2-5)
Numerals†	1 (0-1)	1 (0-2)	0 (0-1)
Women only:	(n=13)	(n=28)	(n=16)
Letters of alphabet*	6 (3-10)	3 (1-6)	3 (1-5)
Numerals†	1 (0-1)	1 (0-1)	0 (0-1)

Fuente: Lyons, R. et al, (1998).

Esto sería tratado por la técnica de OCR con un pre procesamiento de limpieza, además de adoptar una mejor cultura en este sentido para los doctores. Nuevamente, la investigación no tiene como objeto el proceso actual de atención del paciente, sino ofrecer una herramienta para la digitalización y estructuración de la información.

Por otro lado, el problema de los sinónimos o jergas que manejan también es un tema a solucionar. En suma, además de estos sinónimos también se presenta otra dificultad como son las abreviaturas, acrónimos y mala escritura (Perez, A. et al, 2015), la mayoría de estos problemas se resuelven por medio del ingreso de información a la base de datos, para el mejor reconocimiento y clasificación de estos términos, recopilada a través de los mismos doctores.

3.6. Machine Learning

El aprendizaje automático o machine learning es una de las ramas de la inteligencia artificial cuyo objetivo es desarrollar un conjunto de técnicas que permitan a las computadoras poder aprender en base a experiencias. Esto se logra creando algoritmos capaces en detectar los comportamientos y tener la capacidad en base a estos ejemplos reconocer los patrones existentes. (Sancho, F., 2015).

El aprendizaje automático puede entenderse en distintas áreas como en la clasificación. Por ejemplo, en un caso sobre el filtrado de spam, se tiene que examinar si

el e-mail contiene o no información relevante para el usuario, sin embargo, esto puede ser variable y la variabilidad es uno de los problemas que intenta atender machine learning. (Smola, A. & Vishwanathan S., 2008).

Por otro lado, dependiendo de la salida, de acuerdo al tratamiento y las diferentes técnicas utilizadas según la publicación "Introducción al aprendizaje automático" del departamento de ciencias de la computación e inteligencia artificial de la Universidad de Sevilla (Sancho, F., 2015), se puede clasificar en:

- Aprendizaje supervisado: La función generada establece una relación entre entradas y salidas deseadas a partir de una clasificación que ya sabemos.

- Aprendizaje no supervisado: La función generada se forma solo por las entradas al sistema sin conocer su clasificación. Se busca patrones para predecir a las nuevas entradas.

- Aprendizaje semisupervisado: La función generada se forma en una mezcla entre entradas cuyas salidas ya son conocidas y otras que no.

- Aprendizaje por esfuerzo: El algoritmo aprende observando el contexto realizando un proceso de ensayo error y reforzando acciones que tienen respuestas positivas.

- Transducción: La función generada es similar al del aprendizaje supervisado, pero el objetivo es predecir las categorías de nuevos elementos basándose en ejemplos de entradas.

- Aprendizaje multi tarea: Engloba todos los métodos que usan conocimiento previamente aprendido por el sistema a la hora de enfrentarse a problemas parecidos a los ya observados.

Entrando en mayor detalle para la técnica de clasificación, en la presente investigación se han tomado en cuenta la existencia de diversas técnicas que se pueden

utilizar, como se observó en el capítulo anterior. Al realizar un balance acorde a los resultados obtenidos en los trabajos de investigación expuestos en revisión de literatura, se optó por utilizar la técnica de Support Vector Machine pues ha tenido mejor desempeño como se puede apreciar en las siguientes tablas:

Tabla 3.2

Resultados de la investigación con los datos de evaluación

Set	Model	Pr	Re	F1-m
Eval	NB	0.163	0.181	0.131
	DT	0.854	0.851	0.843
	RF	**0.883**	0.881	0.876
	SVM	0.880	**0.889**	**0.878**
Train	NB	0.328	0.394	0.312
	DT	0.905	0.909	0.902
	RF	**0.969**	**0.970**	**0.967**
	SVM	0.959	0.964	0.959

Fuente: Perez, A. et al., 2015

Tabla 3.3 sin expansión de término

Resultados sin expansión de término

	PLAUM	SVM
P	80.91 %	90.48 %
R	64.08 %	61.79 %
F1	71.52 %	**73.43 %**

Fuente: Perea, J. et al., 2009

Como se observa en la tabla 3.2, se presentan los resultados de los indicadores de precisión (Pr), recall (Re) y F1-score (F1-m) tanto en los datos de evaluación como de entrenamiento, estos son 3 indicadores claves que miden el performance de un modelo de predicción o clasificación entre 0 y 100%, y serán definidos a mayor detalle más adelante. Se utilizaron 4 algoritmos para categorización: Naive bayes, Decision Trees, Random Forest, Support vector machine. La técnica de SVM fue la que obtuvo los mejores resultados en los datos de evaluación en la clasificación de términos biomédicos.

Asimismo, en la tabla 3.3, se observa los resultados en un segundo estudio donde la técnica SVM tuvo mejor desempeño en el indicador F1 score para la clasificación de textos médicos.

3.7. K Nearest-Neighbor

También conocido como kNN, es considerado como un método estadístico para realizar reconocimiento no paramétrico como se mencionó anteriormente. Dado su simplicidad y amplio estudio de sus propiedades, es uno de los métodos más comunes para aplicaciones de OCR (Sanchez & Sandonis, 2009). Además, puede llegar a ser casi tan efectivo como otros algoritmos más complejos como la técnica SVM (Delen, 2016), que pueden llegar a tener una diferencia de un 2% a 3% en precisión sin tener en cuenta una optimización de k adecuada (Mohanty & Nandini Das Bebartta, 2011).

El algoritmo también es conocido como uno de aprendizaje basado en instancia y de aprendizaje flojo. La explicación de esto, es que se basa en un conjunto de parámetros sobre los datos que son conocidos, los cuales mantiene en memoria, para posteriormente poder asignar o clasificar nuevos elementos a un conjunto de datos que tienen un mayor parecido. Razón por la cual tiene que recorrer sus datos ya conocidos (Delen, 2016).

En kNN, el k hace referencia al número de vecinos que se usan para la clasificación o regresión, este valor de k se puede escoger según la data que se posea. Además, no existe una forma automatizada para encontrar el k más óptimo, solo se podrá encontrar mediante algún proceso experimental, el cual te permita obtener los mejores resultados puesto que de este dependerá la predicción o resultado (Delen, 2016). Para este algoritmo no existe una etapa de entrenamiento que permita construir un modelo basado en la data como en otros, pero si existe una etapa para encontrar un k óptimo, en el cual se obtenga los mejores resultados para un conjunto de datos determinados (Delen, 2016).

El algoritmo consiste en un conjunto de datos, que también es conocido como objetos prototipo, de los cuales se conoce de forma previa la clase a la que pertenece y otro conjunto de objetos que no tienen una clase definida o conocida. En otras palabras, a partir de imágenes de caracteres que son usados como muestra y caracteres que serán

reconocidos, se tratara de encontrar a los k caracteres más parecidos y dado que estos no poseen una clase se asignará una. Dicha clase, será la que tenga el mayor número de coincidencias entre los objetos prototipo (Sanchez & Sandonis, 2009).

Lo anteriormente descrito corresponde a un kNN de tipo clasificador. Existe también otro tipo que es conocido como de regresión, que a partir de los k objetos prototipo más cercanos a un nuevo objeto de entrada, obtiene la media de sus valores (Sandin, 2015).

Por lo tanto, para ambos tipos de algoritmos, se determinarán las distancias mínimas, entre el dato de entrada y los k objetos prototipo. Los valores de las distancias se pueden determinar a partir de las siguientes fórmulas (Sandin, 2015), elegir entre uno u otro tipo de distancia afecta en la determinación de los vecinos más cercanos (Delen, 2016), algunos son:

- Distancia Euclidiana, mostrada en la ecuación 1, esta distancia es la más conocida y se puede formular a partir del teorema de Pitágoras (Sandin, 2015). Consiste en la raíz cuadrada de la suma de la diferencia elevada al cuadrado, sobre puntos en el espacio euclidiano que tienen como coordenada cartesiana (x_i, y_i).

$$Euclidiana = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \quad (1)$$

- Distancia Manhattan, usa la suma de la diferencia absoluta para determinar la distancia entre vectores que tienen como coordenada cartesiana (x_i, y_i). Esta distancia siempre resultará en mayores valores (Sandin, 2015), dado que toma en cuenta la diferencia absoluta en cada dimensión antes de realizar la suma, como se puede ver en la ecuación 2.

$$Manhattan = \sum_{i=1}^{N} |x_i - y_i| \quad (2)$$

- Distancia Minkowski, es una generalización de los dos anteriores métodos, se muestra en la ecuación 3 (Sandin, 2015). Si q es igual a 2, entonces resulta en la distancia euclidiana o conocida como L2-Norm; si q es igual a 1, entonces se obtiene la distancia Manhattan o L1-Norm.

$$Minkowski = \left(\sum_{i=1}^{N} (|x_i - y_i|)^q\right)^{1/q} \quad (3)$$

El proceso que es recomendado para trabajar con este algoritmo, comienza con encontrar el valor de k y el tipo de distancia que se usará que funcionen mejor, para ello se separara los datos que se tiene actualmente en dos grupos, una data de clasificación y una de validación. Posteriormente, recién se podrá emplear nueva data que no se encuentra clasificada para encontrar a la clase que pertenece (Delen, 2016).

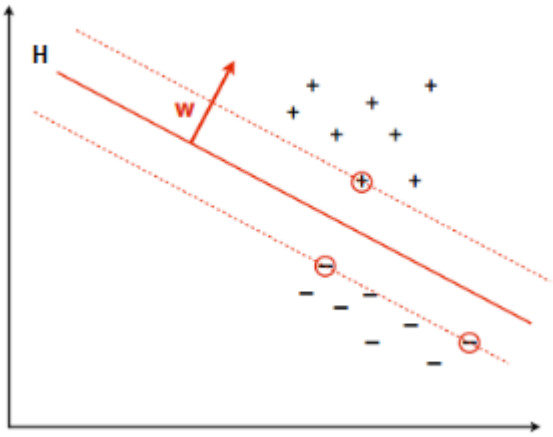
3.8. Support Vector Machine

Support Vector Machine (SVM), o conocido en español como máquinas de soporte vectorial. Support Vector Machines es una técnica de machine learning que se basa en el uso de vectores e hiperplanos para separar y dividir efectivamente conjuntos de datos.

La elección del mejor hiperplano, que separa el conjunto de datos y logra categorizarlos en su respectivo conjunto, se verá influenciado de acuerdo al margen de diferencia que existe entre las distintas clasificaciones. Mientras mayor sea el margen, mayor probabilidad de ser el escogido tendrá (Perez, A. et al., 2015).

Figura 3.5

Clasificación lineal en dos dimensiones



Fuente: Flach, P. (2012)

$$\{x \in S | w \cdot x + b = 0\}, w \in S, b \in R \quad (4)$$

En la figura 3.5 se puede observar como existen dos conjuntos de datos, los puntos positivos y negativos. Siendo en el espacio S, y "x" perteneciendo a dicho espacio, cualquier hiperplano puede ser descrito como la multiplicación entre el vector perpendicular *w* y el vector desde el inicio de coordenadas hasta un punto x más una constante b como se puede apreciar en la ecuación 4 (MIT OpenCourseWare, 2014).

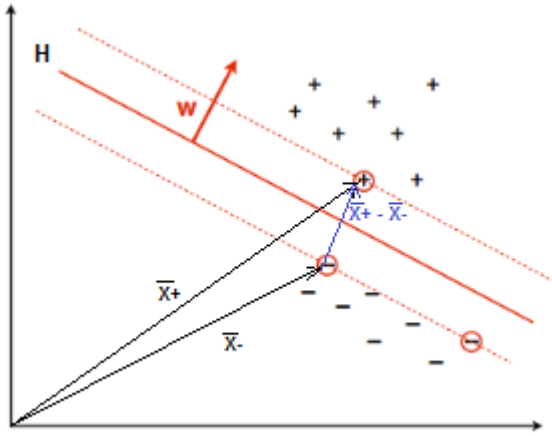
El hiperplano diagonal H define evidentemente las dos regiones de clasificación donde se agrupa un grupo de positivos y otro de negativos. Esta separación lineal dividirá a ambas categorías, donde en la zona de positivos, se definirá como el resultado +1, mientras que en la de negativos -1 como se refleja en la ecuación 5 (MIT OpenCourseWare, 2014):

$$Clase(X_k) = \begin{cases} +1 Si \cdot x_k + b > 0 \\ -1 Si \cdot x_k + b < 0 \end{cases} \quad (5)$$

Utilizando teoría básica de vectores, indica que la distancia entre los puntos fronterizos de cada categoría no es más que la resta de ambos vectores como se refleja en la siguiente figura.

Figura 3.6

Distancia entre ambas categorías.



Fuente: Flach, P. (2012)

El objetivo es ampliar esta distancia del hiperplano (línea diagonal H) entre el grupo de positivos y negativos. Por lo que, la distancia que se desea ampliar no es más que la resta de ambos vectores que están en la frontera de ambos grupos o más cerca al hiperplano.

Sin embargo, un vector no es una distancia numérica, por lo que se deberá pasar a un escalar para obtener la distancia numérica necesaria, para ello se utilizará el vector unitario del vector *w* y se multiplicará por la resta antes mencionada como se aprecia en la ecuación 6 (MIT OpenCourseWare, 2014):

$$D = (x_+ - x_-) \cdot \frac{w}{||w||} \tag{6}$$

Bajo las propiedades aritméticas, se multiplica ambos sumandos por el vector unitario desarrollando la ecuación 6 convirtiéndola en una ecuación 7 (MIT OpenCourseWare, 2014):

$$D = x_+ \cdot \frac{w}{||w||} - x_- \cdot \frac{w}{||w||} \tag{7}$$

Sin embargo, se conoce el producto de x y w dada la ecuación 5, por ello es cuestión de reemplazar valores de 5 en 7 (MIT OpenCourseWare, 2014):

$$D = \frac{(1-b)}{||w||} - \frac{(-1-b)}{||w||}$$
$$D = \frac{2}{||w||} \tag{8}$$

Se ha demostrado que la distancia no es más que el doble de uno sobre el módulo del vector W. Para lograr maximizar el valor de una fracción, se tiene que minimizar el valor del denominador, por lo que se buscaría solo minimizar el valor de: ||w||.

Se sabe que, para lograr minimizar el valor de una función con restricciones, pues el vector W está relacionado como se ve en la ecuación 2 con limitaciones, existen varios métodos matemáticos. Para uso práctico, el método a usar es de los multiplicadores de Lagrange.

Los multiplicadores de Lagrange, es un método para lograr optimizar los valores de una función multivariable sujeta a restricciones (Ivanciuc, O., 2007). Por motivos de simplicidad, es correcto afirmar que minimizar el módulo de W es lo mismo que minimizar el módulo de W al cuadrado sobre dos (MIT OpenCourseWare, 2014). Entonces:

$$L = \frac{||w||^2}{2} - \sum_i \alpha[y_i * (w \cdot x_i + b) - 1] \tag{9}$$

El segundo paso es hallar el extremo de esta función, lo que significa que se tendrá que utilizar derivadas respecto a la variable que se desea derivar, en este caso es w.

$$\frac{\partial L}{\partial w} = w - \sum_i \alpha_i[y_i * (w \cdot x_i)] = 0 \tag{10}$$

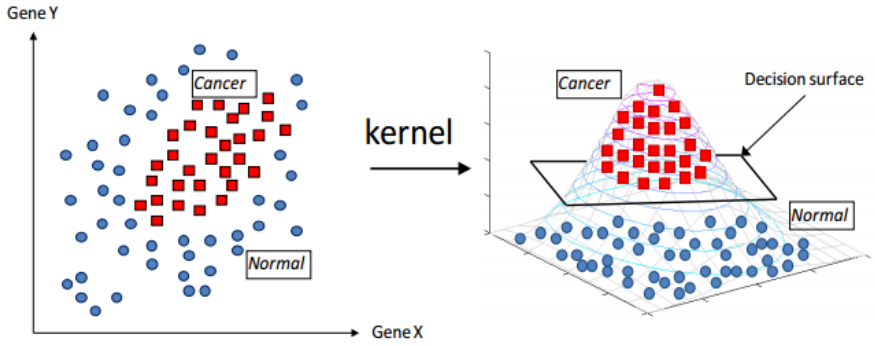
$$w = \sum_i \alpha_i [y_i * (w \cdot x_i)]$$

A partir de ello, se puede concluir que w es una suma lineal de algunas muestras que hay en el espacio, y con algunos referimos a que el alfa será cero. Sin embargo, este es un escenario ideal, por lo que no es aplicable en todos los casos (MIT OpenCourseWare, 2014).

Como se aprecia en la figura 3.7 los puntos cuadrados y los circulares están más dispersos, por lo que la solución no será un hiperplano en un plano bidimensional como se presenta. Aquí entra la función Kernel, para mapear los datos en una dimensión superior mediante una proyección matemática y lograr así separar ambas categorías efectivamente.

Figura 3.7

Función Kernel



Fuente: Statnikov, A. et al., 2009

En este escenario, con este "truco" se puede lograr uno de los objetivos específicos que es la categorización de textos. El "kernel trick" se caracteriza por utilizar funciones kernel. Las funciones kernel permiten convertir un problema de categorización no lineal en el espacio dimensional original a uno lineal en un espacio dimensional mayor. Algunos ejemplos de estas funciones son los siguientes:

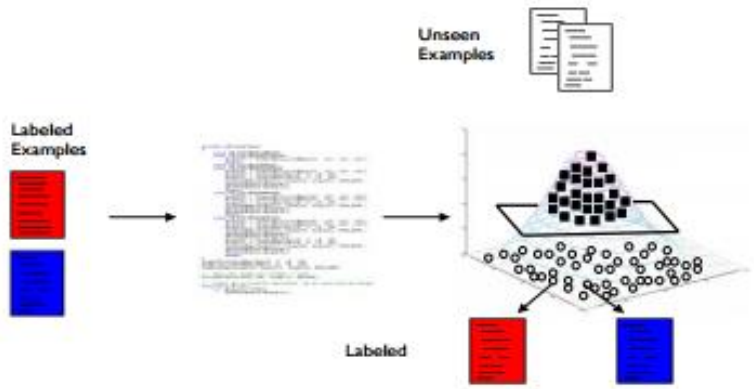
- Kernel Lineal: $\langle x, x' \rangle$

- Kernel Polinomial: $(\gamma\langle x, x'\rangle + r)^d$. Grado (d), Coef (r)
- RBF: $\exp(-\gamma|x - x'|^2)$. Gamma (γ) > 0

En el material de información: “A Gentle Introduction to Support Vector Machines in Biomedicine”, presentan un caso, primero recolectan la información de archivos biomédicos. Luego, representan y asignan pesos según el Metamap y un repositorio. En la tercera fase, como se observa en la figura, una vez identificado los pesos y los labels, se utiliza la técnica SVM para capturar las categorías descritas (Statnikov, A. et al., 2009).

Figura 3.8

Función Kernel aplicada



Fuente: Statnikov, A. et al. (2009)

El cuarto paso es donde se evalúa el performance con validación cruzada y métricas como sensibilidad, especificidad, F1-score, entre otros. Como se ha revisado la siguiente técnica, ha sido aplicada efectivamente en diversos estudios en el campo de clasificación biomédica por lo que se tomará en cuenta para la presente investigación.

3.9. Term frequency – Inverse document frequency

Term Frequency – Inverse document frequency es una medida que refleja tanto la frecuencia de un término (TF) y la frecuencia de documento inversa (IDF) que da un menor peso a los términos más frecuentes entre documentos. (Azure ML, Team, 2015).

La primera métrica (TF) se calcula mediante la fórmula (Mickevicius, V. et al, 2015):

$$tf(t,d) = \sum_{x \in d} fr(x,t) \tag{11}$$

Donde "x" viene a ser una palabra del documento "d" y "t" el termino en cuestión. Además, la función $\mathrm{fr}(\mathrm{x},\mathrm{t})$ evalúa si el término es la palabra del documento, en caso lo sea tomará el valor de 1, así contando la frecuencia del termino en las palabras del documento, en caso que no sea el valor tomará 0. El valor que retornará al final el TF es el número de veces que se repite el término en el documento (Mickevicius, V. et al, 2015).

Por otro lado, el IDF presentaría la siguiente fórmula:

$$idf(t,d,D) = log\frac{N}{1 + |\{d \epsilon D : t \epsilon d\}|} \tag{12}$$

N representa el número de documentos que presenta el corpus, mientras que el valor bajo la división representa el número de documentos en el que figura el término considerando sumarle 1 en caso sea nula la participación del término en los documentos.

Finalmente, TF-IDF solo implicaría una multiplicación vectorial entre ambas matrices (en caso los términos sean más de 2) y/o sea más de un documento. En la ecuación se obtiene la fórmula de dicha métrica (Mickevicius, V. et al, 2015):

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,d,D) \tag{13}$$

TF-IDF es utilizado como métrica en el estudio para evaluar la importancia de un término en documentos. En la presente investigación se considerará en la implementación para evaluar las características de cada categoría, así con estos valores también normalizados se pasará a la clasificación mediante el algoritmo elegido.

3.10. Recursos explorados

Luego de haber explicado las técnicas elegidas, se definieron los recursos comparando nuevamente con los trabajos de investigación anteriores. Para iniciar, el lenguaje de programación escogido fue Python. La justificación se debe a que actualmente, Python es considerada como una de las mejores plataformas junto con R (Auld, A. 2016). Además, dentro de las investigaciones revisadas, el lenguaje utilizado en su mayoría era Java, el cual no posee librerías tan actualizadas en estos campos como Python.

Por otro lado, por parte de la clasificación para las enfermedades se utilizará el ICD-10. El International Statistical Classification of Diseases and Related Health Problems cuyas siglas son ICD que proveen un catálogo de enfermedades o problemas de la salud, este es de mucha utilidad pues en 283 páginas, brinda una clara y numerada clasificación de las enfermedades existentes (World Health Organization, 1993).

Además de ser libre y estar disponible públicamente, se decidió escoger porque se sitúa exactamente en lo que la investigación delimita con respecto al reconocimiento de enfermedades pues ofrece su propia categorización representada en su jerarquía.

Por último, un recurso muy importante es el paquete de trabajo a utilizar. Scikit - learn es un módulo de Python que integra diversos algoritmos de machine learning tales como Random Forest, regresión lineal y multivariante, clusters, árboles de decisiones, Support Vector Machines, entre otros (Pedregosa, F. et al, 2011). Además de tener su propia página web donde tiene documentación y ejemplos donde demuestra su performance, además de un API para descargar en Python, en suma, tiene una gran ventaja debido a que es una herramienta libre y posee el algoritmo de Support Vector Machines con el manejo de Kernels.

La librería a utilizar será NLTK (Natural Language ToolKit). NLTK es un paquete especializado utilizado para desarrollar aplicaciones que trabajen con data del lenguaje humano. (Steven, B. et al, 2009). Además, también incorpora diferentes librerías para la clasificación, tokenización, tagging, parsing, etc. Por estas razones, se decidió finalmente utilizar esta herramienta por su especialización en el campo y además de no ser muy usada en las investigaciones buscadas.

Además, se utilizó la librería “mysqldb” que contiene el driver SQL para Python y así lograr la conexión a la base de datos y poder insertar en las entidades respectivas la información. Se decidió utilizar una base de datos relacional debido a que se quiere estructurar la información para un tratamiento posterior más sencillo y el lenguaje de MySQL por ser libre y reconocido.

En adición a ello, la herramienta es versátil y de código abierto por lo que se tomará el código ya desarrollado y se pasará a la medida para adecuarlo a ciertas métricas y evaluaciones propuestas que pasarán a desarrollarse en la propuesta de solución.

CAPÍTULO IV: PROPUESTA DE SOLUCIÓN

La problemática identificada en el presente trabajo es enfrentada por un sistema que logre digitalizar la ficha de diagnóstico escrita a mano por los doctores y estructurar estos datos. Para lograr lo anterior, se utilizará técnicas de OCR y machine learning, con el fin de la estructuración de los datos, todo este proceso bajo una nueva metodología planteada en un artículo presentado en la revisión de la literatura.

Por otro lado, uno de los beneficios que permitirá la solución planteada, es dar pie al análisis de los datos y generar nuevas oportunidades para la institución clínica. A continuación, se pasará a detallar el alcance, los métodos utilizados y demás detalles de la propuesta de solución.

4.1. Métodos de investigación

El método de investigación es de tipo experimental, puesto que se realizarán distintas pruebas realizando modificaciones al sistema de ser necesario para obtener tasas de precisión aceptables. Durante el desarrollo de la investigación, se aplicaron distintos conceptos aprendidos en el desarrollo de la carrera, reforzados junto a la información obtenida de la revisión de la literatura y del marco teórico.

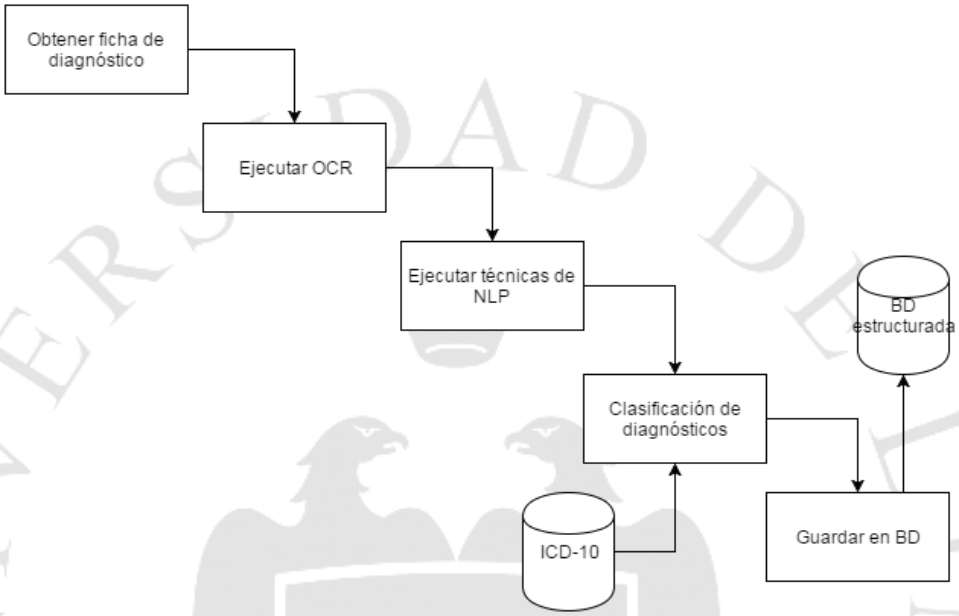
La solución que se plantea, consiste en desarrollar un sistema que permita la estructuración de los datos de fichas de diagnóstico médicas para lo que se seguirá una metodología similar al obtenido de la literatura (Bushinak, AbdelGaber, & AlSharif, 2011). El proceso consiste que una vez que el paciente ambulatorio ha sido tratado por el doctor, este último generará una ficha de diagnóstico según la evaluación que se realice al paciente, posteriormente esta ficha será escaneada.

Una vez con la imagen digitalizada, iniciará la aplicación de técnicas de OCR, la salida de este proceso es un texto digitalizado, el que servirá como entrada para la aplicación de técnicas de NLP. Posteriormente, iniciará la clasificación de diagnósticos

y finalmente será guardado en una base de datos (BD). El anterior proceso se puede visualizar en la figura 4.1.

Figura 4.1

Proceso general de la propuesta de solución.

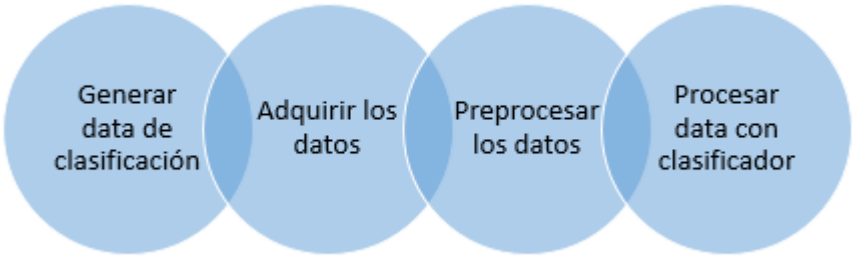


Fuente: Rasmussen, Luke V; et al., (2012)

Para el proceso de OCR, se seguirá un proceso obtenido de la literatura, se escoge este porque permite una arquitectura modular (Rasmussen et al., 2012) para obtener el texto digitalizado a partir de la ficha médica. La modularidad, implica que se podrá cambiar de herramienta o algoritmo de ser necesario durante el transcurso del estudio, esto porque se quiere lograr que el sistema logre una buena precisión. El proceso es como se muestra en la figura 4.2.

Figura 4.2

Proceso utilizado para la digitalización del texto.



Fuente: Rasmussen, Luke V; et al., (2012)

La primera actividad que se realiza durante la propuesta de solución de investigación es un módulo que soportado por un conjunto de software open source, permitan lograr el preprocesamiento de un texto escaneado. En forma concreta, se abarcará la binarización adaptativa de una imagen de la ficha médica, que será lograda con el Iso Data Algorithm (Niklas, 2016) con ayuda de OpenCV que es una librería open source (Landini, 2016).

La siguiente función a tratar será aquella que permita remover ruidos de la ficha de diagnóstico, donde se normalizará la imagen y corregirá las desviaciones (Vamvakas, 2014) con apoyo de OpenCV (Abecassis, 2011). Además, se implementará con un algoritmo para realizar un adelgazamiento de las letras (Guo & Hall, 1989). Posteriormente, se entregará la ficha de diagnóstico segmentada por líneas y caracteres, donde se usarán técnicas de clusterización, donde cada una será dividida en sub imágenes (Vashisht & Nandal, 2016). Adicionalmente, se tendrá una semana donde se podrá realizar las pruebas necesarias para incluir o quitar alguna otra función (Rasmussen et al., 2012).

A continuación, se procederá a realizar el reconocimiento óptico a partir de lo obtenido en la anterior etapa, para ello se hará uso de técnicas no paramétricas, de forma concreta se trabajará con el algoritmo k Nearest-Neighbor (kNN) (Sandin, 2015). La elección de este algoritmo es porque existe bastante literatura acerca de este, es simple y

preciso para realizar la clasificación, justamente por eso también es fácil de implementar. Además, kNN es casi tan preciso como otras técnicas más complejas (Denle, 2016).

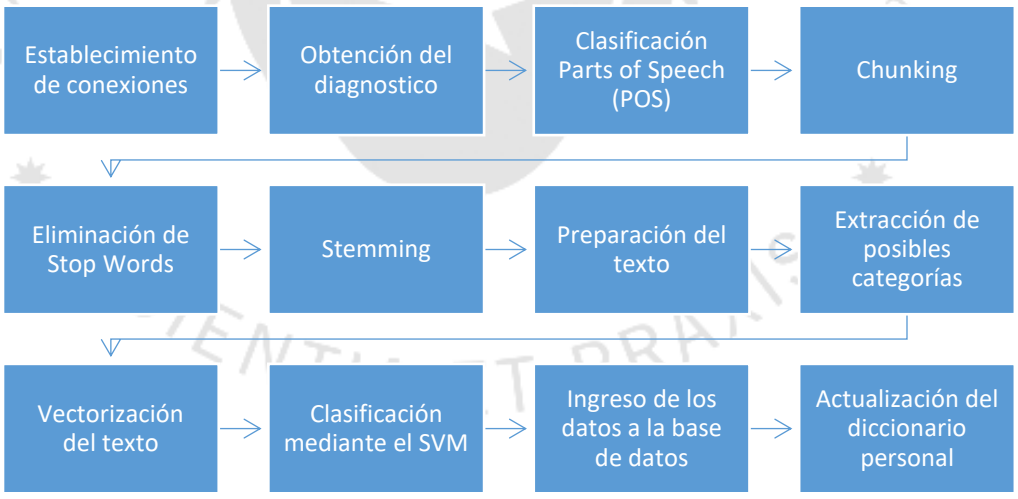
Por otro lado, para realizar el reconocimiento se tendrá que obtener los datos que serán usados para la clasificación (Sandin, 2015), es por ello que se buscará de alguna base de datos libre. Subsecuentemente, se realizará el reconocimiento y las pruebas necesarias, recopilando los resultados que serán útiles para la siguiente etapa.

Como último paso, al menos para esta primera etapa, es realizar el post procesamiento, en la cual se recopilarán todos los resultados obtenidos de la anterior etapa. Adicionalmente, se revisará estos datos tratando de obtener patrones en los datos, esto es conocido como optimización sensible al contexto (CSO) (Rasmussen et al., 2012). Con lo anterior, se obtendrán aquellas palabras que hayan sido mal reconocidas y tratar de obtener las correctas a partir de patrones repetibles.

Luego de haber realizado el OCR se iniciaría el proceso de categorización y estructuración de la información, como se aprecia en la figura 4.3.

Figura 4.3

Proceso de categorización y estructuración.



Fuente: elaboración propia.

El proceso inicia con la captura del texto, la plataforma se integrará con el módulo de digitalización para capturar el texto. Este se recibirá en como parámetro, en caso sea

un archivo plano con extensión “txt”. Además, se extraerá y asignará la cadena de texto a una variable.

Luego de obtener el diagnostico se pasará a clasificar las formas verbales con la clasificación POS (Parts of Speech), la cual permite conocer la estructura de la oración. Además, en este proceso también se ejecutará el chunking y chinking, que ayudarán a limpiar el texto a clasificar. Con el chunking se escogen ciertos patrones de estructuras morfológicas (sustantivo-verbo-sustantivo, por ejemplo) para seguir buscando esas posibles categorías y luego el chinking, que elimina las estructuras morfológicas innecesarias para la clasificación.

Luego se pasará a eliminar las Stop Words, que no son más que palabras que resultan inútiles para el análisis o palabras comunes que se podría obviar para la clasificación. En seguida viene el steamming, que extrae la raíz de la palabra con lo que ayudará a extraer las posibles categorías en la base de la ICD-10. Con lo que resultará en una lista de posibles categorías donde el universo se reduce a la exploración de estas.

Con estas categorías se prepararán a ser vectorizadas para encontrar la categoría más cercana posible, en este pequeño proceso se instancian las variables para ser procesadas. La vectorización del texto es uno de los pasos fundamentales, pues es aquí donde se convierte la cadena de texto en números para lograr crear los vectores y aplicar el algoritmo de Support Vector Machine. La vectorización se logrará mediante un paquete ofrecido por el scikit learn llamado CountVectorizer y TfidfTransformer que convierte una colección de documentos de cadenas de texto en una matriz de cuenta de tokens específicamente utilizando la técnica TF-IDF. (Scikit-learn, 2016).

Además, si no se provee un diccionario y no se usa un analizador que realiza una especie de selección de características, entonces el número de estas será igual al del tamaño del vocabulario encontrado analizando los datos, este paquete ha sido elegido debido a que actualmente ya se cuenta con el ICD-10 que se asimila a un vocabulario con clasificaciones, así también el diccionario personalizado de algunos términos coloquiales del centro para lograr el match deseado para que con lo que el sistema podrá hacer uso y utilizarlo para el CountVectorizer.

En seguida, el siguiente proceso es la implementación de la técnica SVM. El motivo de la elección de este algoritmo se basa en dos principales razones como se mencionan en el artículo "Text Categorization with Support Vector Machines: Learning with Many Relevant Features", estas son:

- La mayoría de problemas de categorización de textos son usualmente separables linealmente: Ohsumed, que es un conjunto de referencias de MEDLINE que consiste en títulos de abstracts y reúne más de trescientos mil trabajos, ha sido separado linealmente en anteriores experimentos. Precisamente, la idea de la técnica SVM es encontrar esa solución lineal (polinomial, RBF, etc.) para lograr categorizar los textos.

- Número de dimensiones que recibe de entrada: En el aprendizaje de los clasificadores de textos se tiene que lidiar con varias características. Uno de los atributos de la técnica SVM es el que dispone de la utilización del sobreajuste, que tiene como fin hacer que el algoritmo de aprendizaje se sobreentrene con los datos de entrada bajo un esquema de aprendizaje supervisado. Esto significa que no dependerá del número de características, más allá de estas tendrá el potencial para lidiar.

Entonces, con el algoritmo ya escogido scikit learn ofrece distintos módulos para trabajar: SupportVectorClassifier (SVC), Nu- SupportVectorClassifier (Nu-SVC) y LinearSVC (Scikit-learn, 2016). Una de las principales diferencias que se puede apreciar entre estos tres módulos es que SVC y Nu-SVC si tienen el atributo de Kernel, mientras LinearSVC no posee de este argumento.

Para este propósito de clasificación multiclase, se utilizará SVC que da mayores oportunidades de optimizar los parámetros, pues a diferencia del Nu-SVC, su parámetro no es binario. Luego de ser clasificada la respectiva categoría podrá ser guardada en un HashMap, una estructura que guarda llave-contenido, donde la llave sería la categoría y el contenido la información extraída.

En esta fase de capturar la categorización, es donde se podrá observar algunas palabras que no tienen una categoría identificada, en esos casos, será ingresado directamente con una etiqueta especial.

Por último, la estructuración de los datos, una vez corregida la categoría del HashMap se pasará a una tabla mediante conexión con base de datos con Python. Se recorrerá cada HashMap con un ID se le asigna al diagnóstico médico y se realiza la transferencia a la base de datos finalizando con los datos ya en una tabla para su futura capitalización.

4.2. Alcance

La finalidad del sistema es estructurar los datos obtenidos a partir de una ficha de diagnóstico escritos a mano con letra imprenta por un doctor. Estos pueden estar disponibles en un formato digital o físico para dejarlos en un estado donde la información pueda ser aprovechada en investigaciones posteriores a esta. Adicionalmente, todo estará incluido en un proceso no intrusivo para el doctor, es decir, el doctor puede efectuar su trabajo sin cambiar su flujo de trabajo actual.

Los documentos a ser escaneados serán aquellas nuevas fichas de diagnóstico generadas después de la investigación. Además, la letra de estas fichas deberá ser imprenta, puesto que la letra corrida presenta un mayor grado de complejidad y otro conjunto de pasos diferentes para realizar el reconocimiento (Plamondon, 2000).

Posteriormente, serán preprocesadas, es decir, se obtendrá una imagen a escala de grises a partir de esta, para luego ser binarizada, se quitarán o reducirá el ruido lo más posible para que la siguiente etapa sea más la efectiva posible. Luego será segmentado y procesado por un algoritmo de reconocimiento de caracteres,

Todos los procesos anteriores, se apoyarán de software open source que justamente permiten realizar cada una de estas etapas. Preferentemente, cada módulo será independiente entre sí para lograr modularidad en el sistema, es decir, se pueden cambiar, agregar, quitar componentes a medida la investigación avance. Además, permitirán poder ser modificados según la solución que se plantea, descartando así alguna solución comercial que pueda imponer algún tipo de restricción.

Si bien se podrán digitalizar registros previos a esta investigación, no se contemplan como parte del alcance. Adicionalmente, sale del alcance de este estudio la digitalización de fármacos recetados al paciente puesto que el foco se da sobre las enfermedades. De la misma forma, sale del alcance mostrar las fichas de diagnóstico digitalizadas al doctor al momento que este realice la consulta.

En seguida, en la categorización de textos, como se mencionó anteriormente, solo se contempla el diagnóstico del paciente, este será categorizado con la técnica de machine learning, Support Vector Machine en base a la información del ICD-10 que se encuentra libre en internet.

Una vez digitalizado el texto, iniciará el pre procesamiento donde se limpiará el texto con respecto a tildes, mayúsculas, minúsculas para que de esta forma los datos queden lo más estándar posible para luego pasar a ser revisada por el responsable del escaneo.

El presente trabajo puede no ser del todo automatizado, pues se quiere buscar el mayor grado de precisión y categorización. Se necesitará de supervisión humana para lograr una eficaz entrada para la categorización en caso hubiese una alta probabilidad de indecisión (Biondich, et al., 2002).

Luego del pre procesamiento, se utiliza el algoritmo modificado de Support Vector Machine para lograr la efectiva categorización. Solo se categoriza el texto del diagnóstico, además se contempla que el sistema solo podrá reconocer el diagnóstico como enfermedades o frases comprendidas en este, mas no podrá categorizar el tipo de medicamento en caso estuviese inscrito en este.

Para el desarrollo del proyecto se utilizarán datos reales provistos por fuentes libres de información que provienen de hospitales o investigaciones previas. Dichos documentos recorrerán por todo el proceso planteado anteriormente para finalmente ser guardados en una base de datos estructurada. Por otro lado, sale del alcance crear un sistema de registros electrónicos puesto que el foco solo es en la estructuración de los datos partiendo de una ficha física de diagnóstico.

4.3. Supuestos

Entre los supuestos de la presente investigación se observaron los siguientes:

- El flujo que seguirá los datos, de inicio a fin será lo más cercano a como sería el funcionamiento del sistema implementado en un ambiente real.

- Para la obtención de los datos se asume que una base de datos libre brindará muestras de fichas de diagnóstico usados en una consulta real.

- Estas fichas de diagnóstico se asumen que son escritas por el doctor en letra imprenta.

- El doctor solo escribirá en la ficha de diagnósticos los diagnósticos relacionados al paciente, mas no algún tipo de medicamento que recomiende.

- Se asume que el doctor permitirá que la investigación tome lugar en su lugar de trabajo, esto incluye la recolección de información acerca del proceso actual de atención del doctor, la modificación en tareas de la secretaria, entre otros factores que se puedan requerir durante la investigación y la validación de este.

4.4. Riesgos

Uno de los riesgos es que el diccionario que se use como léxico para la digitalización no contemple las palabras escritas por el doctor en la ficha de registro, lo que puede resultar en una incorrecta digitalización de la información. Para mitigar esto, se clasificaría el texto en una categoría la cual estará sujeta a revisión manual.

Existe un riesgo asociado a que los algoritmos de reconocimiento de caracteres no puedan reconocer la letra del doctor y se obtenga bajas tasas de reconocimiento. Para mitigar esto, un auto corrector de texto puede ser utilizado y el algoritmo de reconocimiento usará como datos conocidos un conjunto de imágenes de caracteres pertenecientes a la letra de un doctor específico.

Por otro lado, está presente el riesgo en la mala clasificación de los datos. Durante la categorización de los datos del diagnóstico puede ser posible que algunas palabras lleguen a ser mal clasificadas en una categoría no correspondiente.

El riesgo se tiene presente desde el modelamiento del algoritmo, por ello se recurrirá a tratará de cubrir con los mismos doctores ciertas ambigüedades para reducir el error en la categorización. Además, de ser necesario una revisión manual post categorización para los casos que sean muy ambiguos.

El presupuesto asignado para la implementación del proyecto por parte de una clínica no cubriría los costos de implementación, debido a que la investigación requiere de costos iniciales para la instalación, el desarrollo y soporte técnico. Dado que, el presente proyecto no tiene un retorno directo de inversión, el presupuesto asignado puede ser limitado. Por lo que se plantea acordar una reunión con partes interesadas para exponer la propuesta de negocio y negociar el presupuesto.

4.5. Entregables

El presente trabajo se ha manejado con entregables, el primero consiste en las fuentes que usaremos para la obtención de los datos, las cuales tienen que ser lo suficientemente grandes para realizar pruebas u otro requerimiento de los algoritmos a usar.

Con todo lo anterior se definirá una sección de interés en la ficha diagnóstico y comenzará el preprocesamiento, donde se tendrán como entregables:

- La binarización de la ficha médica.
- Entregar ficha médica libre de ruidos.
- Entregar ficha médica segmentada en líneas.
- Segmentar ficha médica en palabras, caracteres.

Una vez que se tenga el texto segmentado empezará la etapa de procesamiento donde los entregables serán tener al algoritmo con la data clasificada y aplicarla al algoritmo para su posterior reconocimiento. La siguiente etapa será la de realizar el post procesamiento donde se tendrá como entregables, encontrar y definir patrones en resultados, además de poder mostrar aquellas palabras que requieran supervisión.

El segundo entregable en esta primera etapa es un prototipo de categorización de textos biomédicos. Principalmente, este con la personalización al algoritmo del API ScikitLearn y en base al recurso de algunas enfermedades del ICD-10 podrá clasificar una serie de textos biomédicos desarrollado en Python. Las funciones serían:

- Recibir el texto biomédico.
- Identificar las enfermedades pertenecientes al ICD-10.
- Categorizar las enfermedades del texto biomédico.
- Estructurar en una pequeña tabla las enfermedades.

En la siguiente etapa, se pretende presentar la implementación del modelo que integrará ambos sistemas tanto el de digitalización con el de categorización y estructuración en una plataforma con las funcionalidades de:

- Obtener una imagen de un texto escaneado.
- Convertir la imagen a una escala de grises.
- Otorgar una imagen libre de ruido.
- Segmentar las palabras.
- Clasificar y realizar el reconocimiento mediante el clasificador kNN para obtener el texto digitalizado.

- Obtener el diagnóstico digitalizado.
- Identificar las categorías del diagnóstico en base a la técnica SVM.
- Categorizar las enfermedades y las partes del diagnóstico.
- Estructurar la información y colocarla en la base de datos.

4.6. Cronograma

Tabla 4.1

	Actividades	Mayo				Junio				Julio			
A	**Realizar Plan de Trabajo de Investigación**	1	2	3	4	1	2	3	4	1	2	3	4
1	Identificación de la idea												
2	Planteamiento del problema												
3	Revision de la literatura												
4	Marco teórico												
5	Propuesta de solucion												
B	**Obtener data**												
1	Obtención de permisos del consultorio												
2	Levantar procesos del consultorio												
3	Obtener registros médicos a digitalizar												
C	**Desarrollo de prototipo de OCR**												
1	Tomar clases online de OCR												
2	Investigar librerías de software correspondientes.												
3	Escanear fichas médicas												
D	**Realizar el preprocesamiento de las fichas médicas**												
1	Binarización de la ficha médica												
2	Entregar ficha médica libre de ruidos												
3	Entregar ficha médica segmentada en líneas												
4	Segmentar ficha médica en palabras, caracteres												
5	Probar distintas combinaciones de tecnicas de procesamiento												
E	**Procesar las fichas con técnicas de OCR**												
1	Entrenar algoritmo												
2	Realizar el reconocimiento												
3	Realizar pruebas												
F	**Realizar postprocesamiento de la ficha médica**												
1	Encontrar patrones en resultados												
2	Definir patrones en resultados												
3	Mostrar palabras sujetas a revision												

Fuente: elaboración propia

Tabla 4.2 Cronograma de agosto a noviembre

Segunda parte del cronograma que comprende los meses de agosto a noviembre 2016.

	Actividades	Agosto				Setiembre				Octubre				Noviembre			
A	**Realizar Plan de Trabajo de Investigación**	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Identificación de la idea																
2	Planteamiento del problema																
3	Revision de la literatura																
4	Marco teórico																
5	Propuesta de solucion																
B	**Obtener data**																
1	Obtención de permisos del consultorio																
2	Levantar procesos del consultorio																
3	Obtener registros médicos a digitalizar																
C	**Desarrollo de prototipo de OCR**																
1	Tomar clases online de OCR																
2	Investigar librerías de software correspondientes.																
3	Escanear fichas médicas																
D	**Realizar el preprocesamiento de las fichas médicas**																
1	Binarización de la ficha médica																
2	Entregar ficha médica libre de ruidos																
3	Entregar ficha médica segmentada en líneas																
4	Segmentar ficha médica en palabras, caracteres																
5	Probar distintas combinaciones de tecnicas de procesamiento																
E	**Procesar las fichas con técnicas de OCR**																
1	Entrenar algoritmo																
2	Realizar el reconocimiento																
3	Realizar pruebas																
F	**Realizar postprocesamiento de la ficha médica**																
1	Encontrar patrones en resultados																
2	Definir patrones en resultados																
3	Mostrar palabras sujetas a revision																

Fuente: elaboración propia

Tabla 4.3 de julio a agosto

Segunda parte del cronograma que comprende los meses de julio a noviembre 2016.

	Actividades	Julio				Agosto				Setiembre				Octubre				Noviembre			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
G	**Desarrollo de prototipo de Categorización**																				
1	Tomar clases de python																				
2	Investigar librerías de software libre en Python de SVM																				
3	Buscar textos médicos para el primer prototipo																				
4	Implementar el SVM en Python para categorizar solo enfermedades																				
H	**Iniciar entrenamiento con el resto del diagnostico**																				
1	Definir más clasificaciones para la data del diagnostico																				
2	Implementar el algoritmo para la categorización																				
3	Entrenar el algoritmo para categorizar el resto del diagnostico																				
I	**Categorizar la data del diagnostico**																				
1	Categorizar toda la data del diagnostico																				
2	Realizar pruebas en la categorización																				
3	Capturar la categorización																				
J	**Estructurar la data en una base de datos**																				
1	Construir el modelo relacional final de la base de datos																				
2	Hacer traspase de Python al modelo relacional																				
3	Realizar pruebas finales																				

Completado
Proyectado

Fuente: elaboración propia

CAPÍTULO V: DESARROLLO DE LA SOLUCIÓN

La presente solución se desarrolló teniendo en cuenta el cumplimiento de los objetivos específicos previamente planteados. A gran escala, la solución consiste en primero digitalizar los datos que posteriormente serán usados por una aplicación. Luego, con los datos clasificados como entrada se aplicará el algoritmo KNN para poder clasificar nuevos caracteres.

El siguiente paso será obtener la información digitalizada para poder, iniciar un preprocesamiento y categorizarla para luego ingresar la información en la base de datos. Una vez completados todos estos pequeños procesos, se habrá finalizado y la data que estaba en papel se habrá estructurado en una base de datos.

Adicionalmente, en el Anexo 8, elaboramos un diagrama de componentes donde se puede ver cómo interactúan los distintos elementos de nuestra solución. Además, en el Anexo 9, hemos listado las librerías utilizadas para el desarrollo de esta solución.

5.1. Digitalización de caracteres

Para la solución de detección de caracteres, se desarrollaron dos aplicaciones en C++ utilizando las librerías de OpenCV usados para la manipulación de imágenes y la implementación del algoritmo de KNN. Ambas aplicaciones son usadas para distintos propósitos; en la primera, se utilizan los datos obtenidos de una base de datos pública para generar archivos que luego, en el segundo proyecto, el clasificador KNN pueda utilizar para poder asignarle una clase a una letra preprocesada. Las aplicaciones se detallan a continuación.

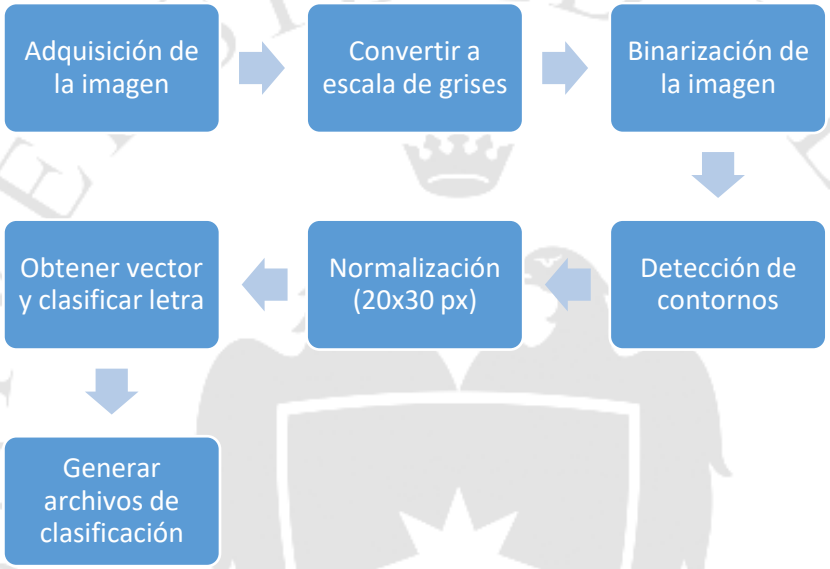
5.1.1. Clasificación de datos de entrada

El flujo de la primera aplicación puede ser resumida en la figura 5.1. En base a ella, la primera aplicación es usada para generar dos archivos que serán importantes para poder

utilizar el clasificador. En esta se puede asignar una clase a cada imagen, es decir, indicar a qué letra del alfabeto, y si es mayúscula o minúscula, corresponde cada imagen.

Figura 5.1

Proceso usado para la primera aplicación para generar un archivo con las clases de cada imagen.



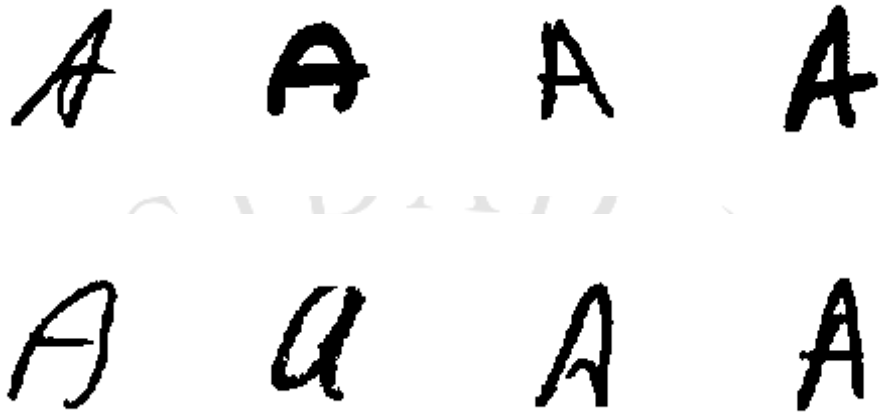
Fuente: elaboración propia.

Como primer paso que sigue la aplicación es la de adquisición de los datos, para ello, se utilizó el NIST Special Database 19 (SD19) (Grother & Hanaoka, 2016). El cual contiene 814255 caracteres escritos a mano, entre los cuales hay letras en mayúscula, minúscula y dígitos, los cuales ya se encuentran segmentados en una imagen de 128 x 128 pixeles.

El SD19 obtiene sus caracteres a partir de formularios y de los cuales se obtuvo muestras de 4169 escritores (Grother & Hanaoka, 2016), esto es importante porque se busca tener clasificado la mayor cantidad de variaciones por cada letra, esto se ve en la figura 5.2.

Figura 5.2

Variedad de letras presentes en el SD19.



Fuente: Grother & Hanaoka (2016).

El NIST SD19 se encuentra organizado por distintas categorías; por escritor, formato, campos de formulario, etc. La clasificación del conjunto de datos que se escogió es aquella basada en clases, es decir, aquella en la que cada letra está organizada por su código en ASCII en hexadecimal.

Es necesario mencionar que para esta solución se tomó como muestra 2000 representaciones por cada letra de los escritores contenidos en la base de datos hsf_0. La razón de este número es porque no todas las letras tienen la misma cantidad de muestras y este número permite tener una muestra balanceada por cada letra. A la misma vez, nos permitirá tener la misma cantidad de datos para realizar las pruebas.

Una vez con los datos adquiridos empieza el preprocesamiento, para ello como primer paso se convierte la imagen a 8 bits, es decir, tendrá una escala de grises cuyos pixeles podrán tomar valores entre 0 a 255. Antes de continuar, es necesario indicar que se aplicó como limpieza de la imagen, una binarización adaptativa e inversión de la escala de grises, como se indicó anteriormente, esto permite resaltar la letra en color blanco y el fondo de negro. Esto se puede ver en la figura 5.3.

Al aplicar ambas técnicas de la forma anterior, cualquier residuo de ruido que haya quedado en la imagen es más notoria por lo que sí quedó un residuo se puede aplicar alguna técnica adicional. Además, es útil para simplificar los cálculos que realice el clasificador puesto que el negro es representado por 0 (Ouchtati, Redjimi, & Bedda, 2015).

Figura 5.3

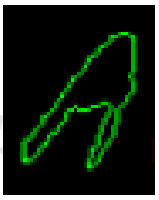
A la izquierda, imagen en escala de grises. A la derecha, imagen binarizada y escala de grises invertida.

Fuente: Elaboración propia.

Una vez con la imagen ya binarizada, siempre y cuando está muestre correctamente la imagen, es decir, no se haya degradado en el proceso, se procede a extraer los contornos más externos de la imagen siempre. Lo anterior se puede ver en la figura 5.4. Es necesario mencionar que solo se extrae el contorno más grande, puesto que en caso de ser menor, algún ruido en la imagen pudo ser marcado como contorno y este no nos interesa, pues puede perjudicar los resultados que obtenga el clasificador.

Figura 5.4 o más externo

Detección de contorno más externo de una letra.



Fuente: elaboración propia.

Al obtener el contorno también permite conocer la posición en la que se encuentra la letra, por lo que podemos dibujar una caja que señale o rodee la región de interés. Dicha zona es aquella con la que se trabajara para los siguientes pasos, esto evitará trabajar con

la imagen original y centrar mayor atención en los pixeles donde solamente aparece la letra.

La zona marcada, a partir de la imagen binarizada, es extraída y guardada en otra imagen la cual será usada para cambiar el tamaño de la imagen que contiene a la letra a un tamaño de 20x30 pixeles (Pradeep, Srinivasan, & Himavathi, 2011). El anterior valor es escogido porque se demostró que a partir de una medida de 20x20 pixeles aumenta la precisión (Lei He, Ping, Jianxiong, Ching, & Tien, 2012). El flujo anterior se puede ver en la figura 5.5.

Figura 5.5

De izquierda a derecha, región de interés marcada en la imagen original, imagen binarizada la cual será redimensionada, imagen redimensionada a 20x30 pixeles.

Fuente: elaboración propia.

Una vez que se obtuvo la imagen a ser usada, se tendrá que asignar una clase, es decir, se indicará mediante un código ASCII decimal a qué letra del alfabeto corresponde y si es minúscula o mayúscula. Además de ello, debido a cómo funciona la librería OpenCV, de forma específica cómo funciona el algoritmo de kNN, se tienen que trabajar con vectores y dado que la imagen representa un conjunto de pixeles en forma bidimensional tiene que ser convertida a una matriz unidimensional o vector.

Por lo tanto, ambos, el código ASCII y la representación de la imagen serán almacenados en matrices diferentes los cuales a su vez serán guardados en diferentes archivos. Sin embargo, las filas de estas matrices serán tratadas como vectores, de esta forma el primer código ASCII de la letra en el vector corresponde a la primera fila de la otra matriz que contiene los valores de cada pixel.

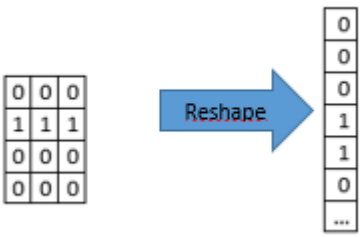
Como ejemplo de lo anterior, “A” que tiene como código ASCII decimal es 65 es guardada en una matriz y la imagen tiene que ser convertida o reorganizada en un vector el cual contiene el valor de los pixeles para después ser guardada en otra matriz.

Los archivos generados están en formato XML, consisten de lo siguiente; el primer archivo, el cual básicamente es un vector, puesto que está conformado por una sola columna y en cada fila contiene el valor ASCII de cada letra, es decir, su clase. Lo anterior se puede en la figura 5.8.

El segundo archivo generado es una matriz de tamaño igual al número de caracteres clasificados por el número de píxeles de cada imagen. Se puede ver en la figura 5.9. Debido a que todas las imágenes son manejadas en un tamaño de 20x30 pixeles, el número de columnas de la matriz será de 600 pixeles, esto porque cada imagen se guarda como un vector esto se puede ver en las figuras 5.6 y 5.7. Por lo tanto, cada fila está asociada a una letra y contiene los valores que toman los pixeles.

Figura 5.6

Reorganización de la imagen en un vector.



Fuente: elaboración propia.

Figura 5.7

La letra A (arriba) es reorganizada en un vector de 600 pixeles de largo (abajo).

Fuente: elaboración propia.

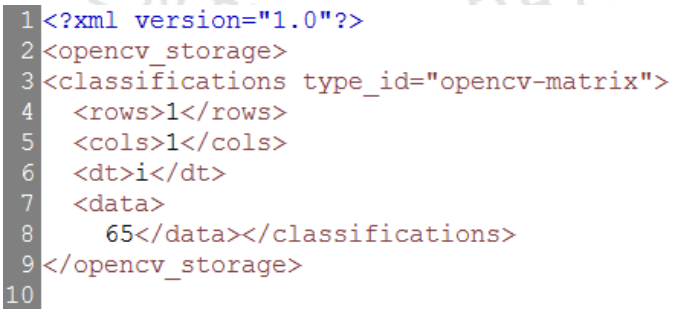
Figura 5.8

Contenido del archivo que contiene los valores de cada uno de los pixeles de una letra A.

Fuente: elaboración propia.

Figura 5.9

Archivo que contiene el código ASCII decimal de una letra A.

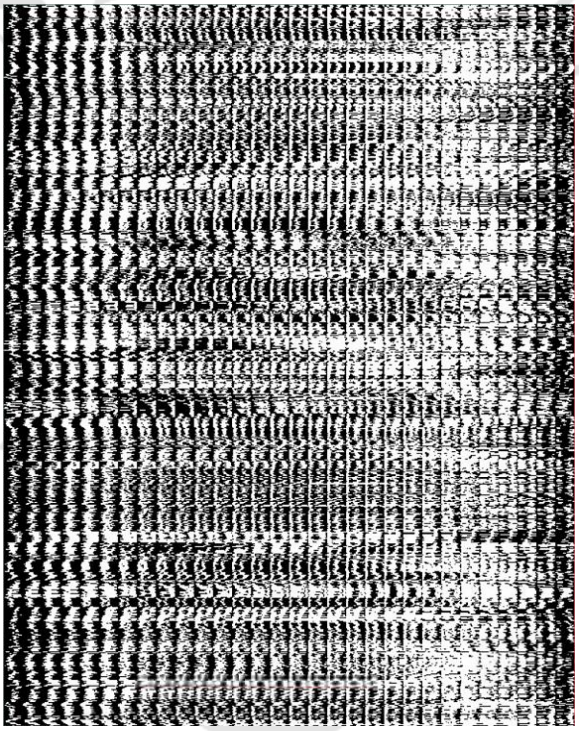


Fuente: elaboración propia.

El segundo archivo que se genera se forma a partir de la agregación o aplicación de los vectores, formando la siguiente figura. Como se mencionó anteriormente, no se debe trabajar con este archivo como imagen en sí, sino como un conjunto de vectores puesto que cada entrada en esta matriz se encuentra asociada a una clase. En sí, la figura 5.10, vendría a ser un conjunto de líneas de la figura 5.7.

Figura 5.10

Representación gráfica de un conjunto de 2000 vectores.



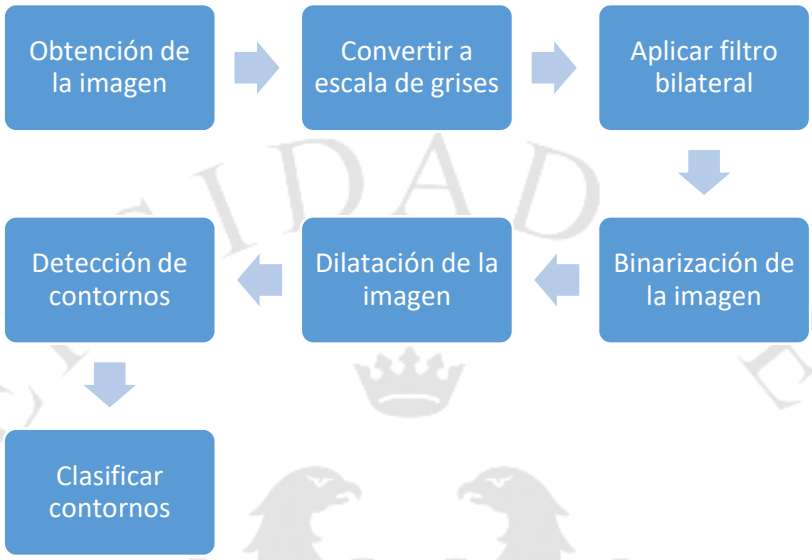
Fuente: elaboración propia.

5.1.2. Reconocimiento de caracteres

El segundo proyecto que se realizó consistió en implementar el clasificador KNN para reconocer las letras de una palabra que contenga letras diferentes a las que clasificadas. El proceso en mención se encuentra graficado en la figura 5.11.

Figura 5.11

Proceso aplicado para el segundo proyecto con el fin de clasificar los contornos que contienen letras.

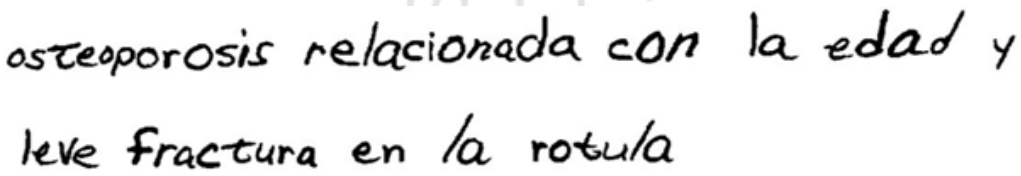


Fuente: elaboración propia.

Por ello, se realizó como primer paso obtener los dos archivos previamente generados, que contienen las imágenes clasificadas, las cuales serán almacenadas en memoria para su posterior uso por el clasificador. Luego, se pasa a obtener la imagen a ser escaneada, en este caso se formó un texto conformada por imágenes obtenidas del conjunto de datos Nist Special Database 19.

Figura 5.12

Las palabras están conformadas por letras no contenidas en la base de datos previamente clasificados.



Fuente: elaboración propia.

La figura 5.12 representa un ejemplo de texto que recibiría la aplicación. Esta imagen posteriormente será preprocesada, por lo que se convierte esta imagen a una escala de grises de 8 bits. Luego se aplica un filtro bilateral y pasa directo a la binarización adaptativa e inversa de la imagen, por las mismas razones que se explicaron anteriormente. El resultado se puede ver en la figura 5.13.

Figura 5.13

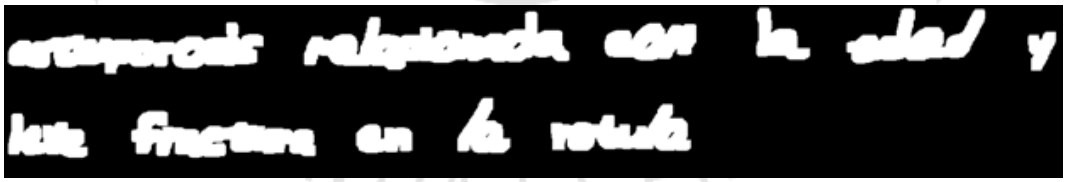
Imagen después de ser limpiada y binarizada

Fuente: elaboración propia.

Como siguiente paso se procede con la detección de palabras, por lo que se dilata la imagen un número determinado de veces hasta que las letras que estén contiguas se agrupen en lo que se podría describir como una gran mancha blanca. Esta, para nosotros significa que en esa región existe una palabra y obviamente hay letras por reconocer. Lo anterior se ve claramente en la figura 5.14.

Figura 5.14

Imagen luego de pasar por un proceso de dilatación.



Fuente: elaboración propia.

Posteriormente, se extraen los contornos más exteriores de la imagen, para detectar donde se encuentran estas palabras. Esto se puede ver en la figura 5.15. Luego, utilizando las coordenadas que cada contorno posee dentro de la imagen, estos fueron ordenados de izquierda a derecha y arriba hacia abajo. Por la cual, se puede obtener el texto en secuencia, como si fuese leído por una persona.

Por otro lado, se descartó realizar una segmentación de líneas utilizando algún método como la transformada de Hough o utilizando una proyección de histograma horizontal puesto que se optó por utilizar el método anteriormente explicado, el cual es más rápido de implementar y permitió lograr la misma funcionalidad.

Figura 5.15

Detección de contornos en una oración y en una palabra.

Fuente: elaboración propia.

Para este momento, ya se conoce donde se encuentra cada palabra, entonces se procederá con el procesamiento ordenado de cada letra por cada palabra, para lo que se extraerán varios contornos por cada palabra, no solo uno como en el anterior proyecto. Esto se puede ver en la figura 5.16. Además, estos contornos son ordenados de izquierda a derecha para que de esa forma se haga el reconocimiento de cada letra y se obtenga la misma secuencia de caracteres que conforman la palabra.

Figura 5.16

A la izquierda, contornos detectados; a la derecha, regiones de interés.

Fuente: elaboración propia.

Adicionalmente, se realiza una validación acerca del tamaño del área de cada contorno, puesto que no tiene que ser muy pequeño para ser considerado como una letra y no un posible ruido no eliminado de la imagen. Con lo anterior, se puede enmarcar los contornos de las letras conocidas, las cuales representan la zona de interés.

Una vez obtenidos los contornos y como ya se conocen las zonas de interés, se extrae de la imagen cada letra y se normaliza a un tamaño de 20x30 pixeles, como se hizo con la data clasificada en el anterior proyecto. Entonces para cada una de estas imágenes, se reorganiza la matriz que la representa en un vector, el cual será almacenado en una matriz.

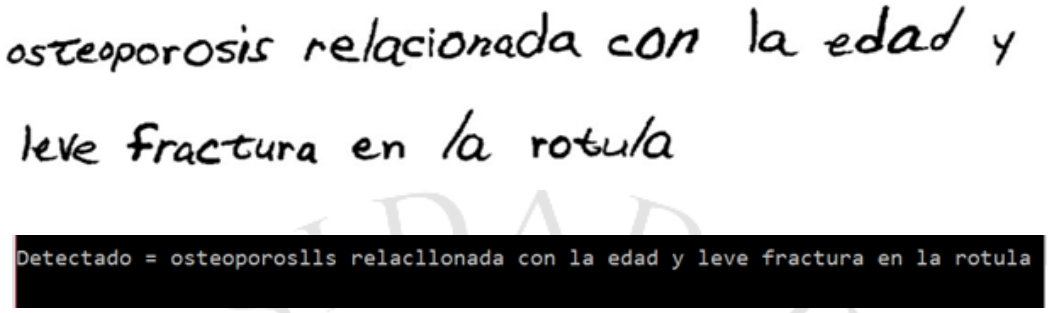
Todo lo anterior se realiza, para que tenga el mismo formato del conjunto de datos que fueron previamente clasificados. Una vez que se tiene los datos en dicho formato recién pueden ser usados por el clasificador.

Al encontrarnos ante un problema de clasificación el algoritmo determinará por votación cuáles son los vecinos más cercanos. Además, para los vecinos calculará la distancia con el vector no clasificado y los ordenará de acuerdo a esta distancia. Con lo anterior, el clasificador kNN podrá encontrar la letra que más se asemeje, es decir, se le asignará una clase, esto se repetirá por cada letra para finalmente obtener las palabras dadas.

La figura 5.17 es ejemplo de lo anteriormente descrito. Es necesario aclarar que en esta figura, las palabras fueron ordenadas de tal forma para demostrar que se puede separar en palabras y reconocer entre líneas.

Figura 5.17

Resultado obtenido por el clasificador al procesar una oración.



Fuente: elaboración propia.

5.1.3. Pruebas y validación de reconocimiento de caracteres

Según la literatura revisada, se plantea el siguiente esquema para las pruebas y validación del reconocimiento de caracteres: como primer paso es definir qué es lo que se iba a digitalizar, en este caso se escogieron las fichas médicas o formatos que representan estos (Biondich, et al., 2002).

En seguida se establece un gold standard (Rasmussen et al., 2012), el cual consiste en realizar un conteo de todas las palabras, campos o regiones a digitalizar. Además, realizar una revisión manual de estos e indicar cuál es su valor, tarea que puede desempeñar dos o más personas (Biondich, et al., 2002). Inclusive, se puede agregar a un mediador (Rasmussen, et al., 2012), para asegurar que se mantenga consistencia en los resultados.

Una vez con lo anterior, se procede a digitalizar los registros y se podría realizar un doble procesamiento (Biondich, et al., 2002) para asegurar la consistencia del clasificador. Con el clasificador, se puede establecer un nivel de confianza y de precisión, el primero se puede obtener a partir de la distancia de los elementos a clasificar con sus respectivas clases. El segundo, a partir de cuantos resultados correctos se pueden obtener del total de elementos a escanear. Con estas pruebas, es donde las configuraciones del pre procesamiento pueden variar (Rasmussen, et al., 2012).

5.2. Categorización y estructuración de la información

Una vez digitalizada la información se plantea dos subprocesos para poder categorizar y estructurar la información. Antes que nada, mediante la librería NLTK se podrá realizar el preprocesamiento del texto más sencillo por los métodos que ya tiene, en estos se plantean diferentes estrategias que se explicarán a detalle en los siguientes párrafos para lograr obtener lo más relevante del diagnóstico y facilitar su clasificación.

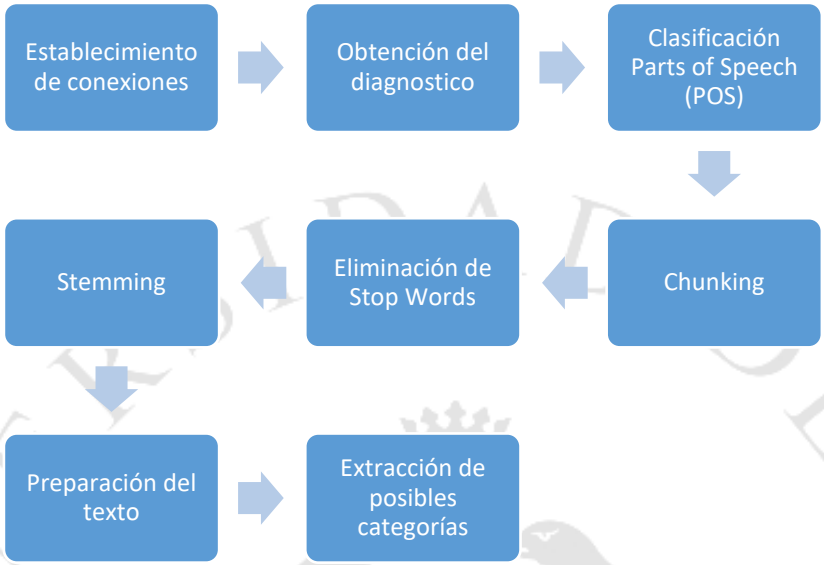
Una vez que se tenga el texto relevante, se iniciará el proceso de clasificación y estructuración mediante el paquete de ScikitLearn con la técnica SVM. Finalmente, una vez categorizada la información, se pasará a ingresar a la base de datos culminando la tarea. En este punto, se podrá decir que se logró terminar todo el proceso logrando pasar los datos que estaban inicialmente en papel y no estructurados a una base de datos estructurada para su futura explotación y análisis.

5.2.1. Pre procesamiento de la información

En la siguiente figura se esquematiza los pasos principales del preprocesamiento de la información:

Figura 5.18

Preprocesamiento de la información



Fuente: elaboración propia.

El establecimiento de conexiones es el primer paso, pues se tiene una base de datos MySQL con la cual se interactuará durante todo el proceso y al final donde se albergará la información estructurada. En esta base de datos se albergan las tablas:

- Catálogo de enfermedades ICD-10: Contiene todas las enfermedades que tiene registrada la International Statistical Classification of Diseases and Related Health Problems cuyas siglas son ICD, en este caso se tiene la versión 10. Tiene como ID el código de enfermedad registrada por la ICD-10, la descripción de la enfermedad, el idFamilia que es el identificador en el catálogo de familia de enfermedades que son los 3 primeros caracteres del idEnfermedad y por último la descripción real, puesto que la primera descripción es una descripción procesada para su fácil análisis.

Figura 5.19

Vista de algunos registros de la tabla “catalogo”.

idEnfermedad	descripcionEnfermedad	idFamilia	descripcionReal
T490	ENVENENAMIENTO AGENTES TOPICOS AFECTAN PRI...	T49	ENVENENAMIENTO POR AGENTES TOPICOS QU...
T494	ENVENENAMIENTO AGENTES TOPICOS AFECTAN PRI...	T49	ENVENENAMIENTO POR AGENTES TOPICOS QU...
X612	ENVENENAMIENTO AUTOINFLIGIDO INTENCIONALME...	X61	ENVENENAMIENTO AUTOINFLIGIDO INTENCIO...

Fuente: elaboración propia.

- Familia de enfermedades: A diferencia de la anterior, esta tabla contiene la descripción general de la enfermedad en la versión pre y pos procesada. En el listado de la ICD-10, cada enfermedad tiene una categoría, estas categorías son las que están en esta tabla y servirán para tener la llave general de la clasificación.

Figura 5.20

Vista de algunos registros de la tabla “familia”.

idFamilia	descripcion	limpio
X61	ENVENENAMIENTO AUTOINFLINGIDO INTEN...	ENVENENAMIENTO AUTOINFLINGIDO INTENCIONAL...
Y83	CIRUGIA Y OTROS PROCEDIMIENTOS QUIR...	CIRUGIA PROCEDIMIENTOS QUIRURGICOS CAUSA ...
Y84	OTROS PROCEDIMIENTOS MEDICOS COMO ...	PROCEDIMIENTOS MEDICOS CAUSA REACCION AN...
Y56	EFECTOS ADVERSOS DE AGENTES TOPICOS...	EFECTOS ADVERSOS AGENTES TOPICOS AFECTAN P...

Fuente: elaboración propia.

- Repositorio de diagnósticos: La siguiente tabla ya llega a ser el destino final de todos los datos ingresados, desde la fecha donde se ingresó el diagnóstico, el id, la misma descripción del diagnóstico y la enfermedad con la cual fue categorizada.

Figura 5.21

Vista de algunos registros de la tabla “diagnóstico”.

	idDiagnostico	diagnostico	enfermedades	fechaDeAlta
	20160901092010000	Osteoporosis relacionada c...	S82	01/09/2016

Fuente: elaboración propia.

El siguiente proceso es la obtención del diagnóstico. Se tiene pensado que la digitalización se hará por cada registro y cada uno tendrá una salida que será un archivo

plano de texto. Entonces, el sistema de la presente investigación obtendrá uno a uno y el texto lo instanciará como una variable para empezar el próximo proceso.

La clasificación POS (Parts of Speech) refiere a una clasificación mediante la librería nltk: StanfordPOSTagger. En el siguiente ejemplo se aplica con el diagnóstico de prueba: “Osteoporosis relacionada con la edad y leve fractura en la rótula, reingresante.”, el tagger lo que utilizó fue un diccionario y con ello clasificó como sustantivos (‘nc…’), determinantes (‘da…’), adjetivos (‘aq…’), etc. Este Tagger es uno utilizado por su alta efectividad a la hora de clasificar cada palabra como se ha observado en la revisión de literatura.

Figura 5.22

Clasificación como tipo de palabra

```
[('osteoporosis', 'nc0n000'), ('relacionada', 'aq0000'), ('con', 'sp000'), ('la', 'da0000'),
('edad', 'nc0s000'), ('y', 'cc'), ('leve', 'aq0000'), ('fractura', 'nc0s000'), ('en', 'sp000'),
('la', 'da0000'), ('rotula', 'aq0000'), (',', 'fc'), ('reingresante', 'aq0000')]
```

Fuente: elaboración propia

Con las etiquetas asociadas a cada palabra se inicia con el “Chunking” y “Chinking”. Una vez clasificada cada palabra en cuanto a su tipo (verbo, sustantivo, preposición, determinante, etc.) se buscará crear patrones que ayuden a entender mejor el contexto para determinar precisamente la oración con los tags que brinda el StanfordPOSTagger.

Figura 5.23

Tags del Stanford POSTagger

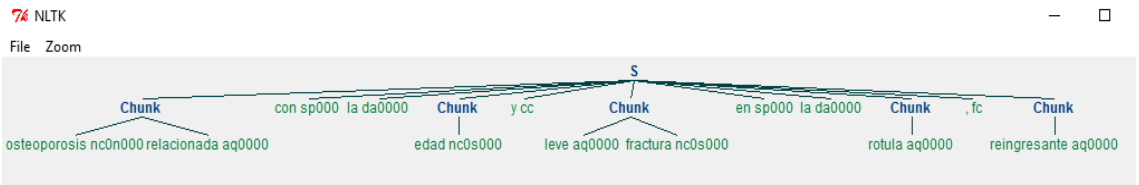
Tag	Description	Example(s)
Adjectives		
ao0000	Adjective (ordinal)	*primera, segundo, últimos*
aq0000	Adjective (descriptive)	*populares, elegido, emocionada, andaluz*
Conjunctions		
cc	Conjunction (coordinating)	*y, o, pero*
cs	Conjunction (subordinating)	*que, como, mientras*
Determiners		
da0000	Article (definite)	*el, la, los, las*
dd0000	Demonstrative	*este, esta, esos*
de0000	"Exclamative" (TODO)	*qué (¡Qué pobre!)*
di0000	Article (indefinite)	*un, muchos, todos, otros*
dn0000	Numeral	*tres, doscientas*
do0000	Numeral (ordinal)	*el **65** aniversario*
dp0000	Possessive	*sus, mi*
dt0000	Interrogative	*cuántos, qué, cuál*

Fuente: The Stanford Natural Language Processing Group (2017)

Por ejemplo, en el caso anterior, en el diagnóstico se tenía la palabra fractura, entonces utilizando el patrón de (Sustantivo y/o conjunción y/o adjetivo). En el árbol generado se observa: “osteoporosis relacionada”, “edad”, “leve fractura”, “rótula” y “reingresante”.

Figura 5.24

Diagrama de chunking definido



Fuente: elaboración propia

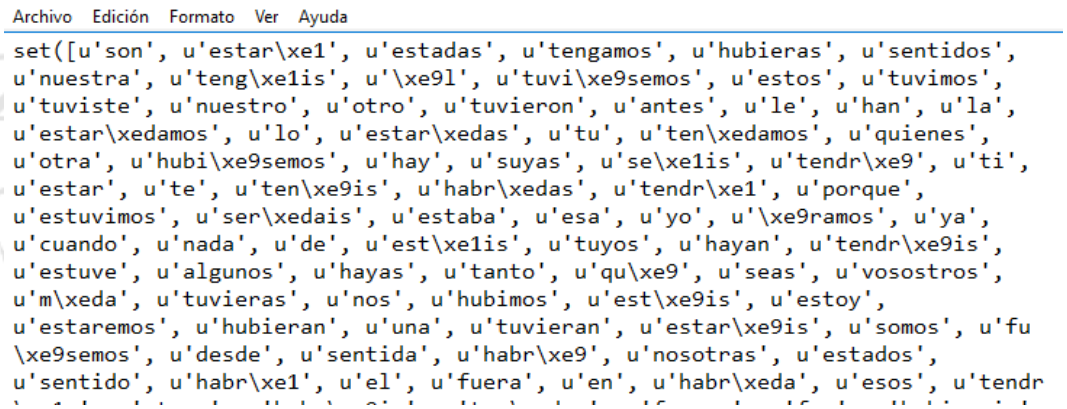
Como se observa, con estos grupos preliminares de palabras se va a separar los que servirán para una nueva revisión en la base de datos y asegurar tener mayor precisión

en la obtención de las categorías posibles. Con estas categorías posibles se pasará al proceso de categorización con la técnica SVM.

En seguida, se pasará un último filtro para tener los grupos limpios para la nueva revisión. Este es el eliminar las Stop Words, que como se describe en el capítulo IV, no son más que palabras que no resultan competentes para un efectivo análisis. NLTK contiene ya un extracto de palabras, sin embargo, gracias al entrenamiento con los datos de MedLine se pudo hallar unas palabras que era frecuentes que no aportaban a la categorización, a lo que se llamará Common words. Entonces, tanto las "stop words" como "common words" fueron separadas para no generar ruido, en la siguiente figura se observa alguna de estas palabras que trae el paquete NLTK en stop words sin un post procesamiento:

Figura 5.25

Ejemplo de "stop words" almacenadas en NLTK



Fuente: elaboración propia

Una vez obtenido los grupos para la evaluación de posibles categorías, se pasará a realizar el Stemming. En la siguiente imagen se puede apreciar alguno de los criterios que se utilizan para hallar las raíces ya definidos por el NLTK.

Figura 5.26

Ejemplo de sufijos almacenados en NLTK

Archivo Edición Formato Ver Ayuda

```
__vowels = "aeiou\xE1\xE9\xED\xF3\xFA\xFC"
    __step0_suffixes = ("selas", "selos", "sela", "selo", "las",
                        "les", "los", "nos", "me", "se", "la", "le",
                        "lo")
    __step1_suffixes = ('amientos', 'imientos', 'amiento', 'imiento',
                        'aciones', 'uciones', 'adoras', 'adores',
                        'ancias', 'log\xEDas', 'encias', 'amente',
                        'idades', 'anzas', 'ismos', 'ables', 'ibles',
                        'istas', 'adora', 'aci\xF3n', 'antes',
                        'ancia', 'log\xEDa', 'uci\xf3n', 'encia',
                        'mente', 'anza', 'icos', 'icas', 'ismo',
                        'able', 'ible', 'ista', 'osos', 'osas',
                        'ador', 'ante', 'idad', 'ivas', 'ivos',
                        'ico',
                        'ica', 'oso', 'osa', 'iva', 'ivo')
    __step2a_suffixes = ('yeron', 'yendo', 'yamos', 'yais', 'yan',
                         'yen', 'yas', 'yes', 'ya', 'ye', 'yo',
                         'y\xF3')
    __step2b_suffixes = ('ar\xEDamos', 'er\xEDamos', 'ir\xEDamos',
                         'i\xE9ramos', 'i\xE9semos', 'ar\xEDais',
                         'aremos', 'er\xEDais', 'eremos',
                         'ir\xEDais', 'iremos', 'ierais', 'ieseis',
                         'asteis', 'isteis', '\xE1bamos',
                         '\xE1ramos', '\xE1semos', 'ar\xEDan',
```

Fuente: elaboración propia

Con estos sufijos y prefijos ya definidos se procesa cada palabra donde por ejemplo se obtiene la relación:

Figura 5.27 er ming en 6 palabras

Aplicación de stemming en 6 palabras

```
['osteoporosis', 'relacionada', 'leve', 'fractura', 'rotula', 'reingresante']
['osteoporosis', 'relacion', 'lev', 'fractur', 'rotul', 'reingres']
```

Fuente: elaboración propia

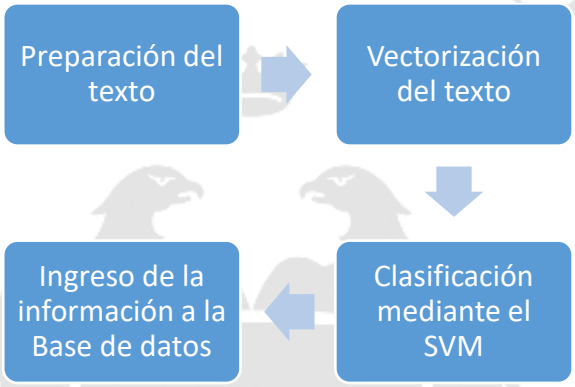
Por ejemplo, como se puede apreciar: “relacionada” ha sido cortada a su lexema “relación” o “leve” ha pasado a ser “lev”. Entonces, luego de haber elegido las palabras relevantes y haber extraído la raíz de estos se consultará a la base de datos por las posibles enfermedades. Con las siguientes seis palabras se pasará a formar parejas, de este modo, de las 12 mil enfermedades que existen en la base de datos, seleccionará aquellas que se asemejen a dicho texto.

5.2.2. Categorización y estructuración

Luego de haber realizado el preprocesamiento de la información, sigue el proceso de categorización y estructuración de la información como se esquematiza en la siguiente figura:

Figura 5.28

Categorización y estructuración de la información



Fuente: elaboración propia

Una vez con el texto preprocesado, se tiene que preparar para iniciar la categorización. En este caso, con las palabras clave mapeadas y las estructuras claves del chunking se procederá a instanciar un arreglo donde se guarde este conjunto de elementos. Gracias al paquete del Scikit learn se instanciará e iniciará el proceso de vectorización.

La vectorización del texto y la clasificación con la técnica de SVM son dos procesos que van en conjunto en la implementación. La vectorización es el proceso principal dentro del cual los tokens en el documento se convierten en números a una matriz, como, por ejemplo, el vector TF-IDF que utiliza el contexto para evaluar la relevancia de una palabra en una colección de documentos. En este proceso se destaca el uso del Scikit Learn con el paquete Vectorizer junto al TfidfTransformer. Como se visualiza en la siguiente figura:

Figura 5.29

Resultados TF-IDF

```
Test set:
problemas relacionados con abuso sexual nino persona dentro
Vocabulary:
{'relacionados': 1, 'eventos': 3, 'rotula': 9, 'perdida': 5
Testeando la matriz de freq
matrix([[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
TF-IDF:
TfidfTransformer(bm25_tf=False, delta_idf=False, norm='l2',
         sublinear_tf=True, use_bm25idf=False, use_idf=True
Con la Matriz de test (IDF):
array([ 7.83733281,  7.83733281,  7.83733281,  7.83733281,
        7.83733281,  7.83733281,  7.83733281,  7.83733281,
```

Fuente: elaboración propia

El test set viene a ser el diagnóstico, como se aprecia en las primeras líneas de la figura 5.29, y el vocabulario contiene las características del diagnóstico. En base a la explicación dada anteriormente del TF-IDF, los términos serían los presentes en el vocabulario y los documentos las enfermedades en dicha categoría. Por ejemplo, al ser 4 términos (características del diagnóstico) y 2 documentos (posibles enfermedades) el TF-IDF resulta ser una matriz 2x4, como se observa en la segunda matriz de la figura 5.30.

Figura 5.29

Multiplicación matricial

$$\begin{bmatrix} \mathrm{tf}(t_1,d_1) & \mathrm{tf}(t_2,d_1) & \mathrm{tf}(t_3,d_1) & \mathrm{tf}(t_4,d_1) \\ \mathrm{tf}(t_1,d_2) & \mathrm{tf}(t_2,d_2) & \mathrm{tf}(t_3,d_2) & \mathrm{tf}(t_4,d_2) \end{bmatrix} \times \begin{bmatrix} \mathrm{idf}(t_1) & 0 & 0 & 0 \\ 0 & \mathrm{idf}(t_2) & 0 & 0 \\ 0 & 0 & \mathrm{idf}(t_3) & 0 \\ 0 & 0 & 0 & \mathrm{idf}(t_4) \end{bmatrix}$$
$$= \begin{bmatrix} \mathrm{tf}(t_1,d_1) \times \mathrm{idf}(t_1) & \mathrm{tf}(t_2,d_1) \times \mathrm{idf}(t_2) & \mathrm{tf}(t_3,d_1) \times \mathrm{idf}(t_3) & \mathrm{tf}(t_4,d_1) \times \mathrm{idf}(t_4) \\ \mathrm{tf}(t_1,d_2) \times \mathrm{idf}(t_1) & \mathrm{tf}(t_2,d_2) \times \mathrm{idf}(t_2) & \mathrm{tf}(t_3,d_2) \times \mathrm{idf}(t_3) & \mathrm{tf}(t_4,d_2) \times \mathrm{idf}(t_4) \end{bmatrix}$$

Fuente: Christian S. (2011)

En seguida, se ingresa la matriz para una categorización mediante la técnica de SVM que indica tanto el hiperplano como la categoría asignada. Para la clasificación se inicia entrenando con las enfermedades de la categoría general en la base de catálogo y el corpus de la ICD con las subcategorías específicas de cada familia en la base de enfermedades. Por ejemplo, en esta ocasión se obtuvo dos posibles categorías:

Figura 5.30

Posibles categorías obtenidas

```
6. Clasificacion SVM
0 problemas relacionados con eventos llevaron perdida autoestima infancia
1 fractura rotula
```

Fuente: elaboración propia.

La categoría 0 indica “problemas relacionados con eventos llevaron perdida autoestima infancia”, mientras que la categoría 1 “fractura rotula”. Luego de haber clasificado se obtendrá la categoría ideal de enfermedad que se capturará en una variable para ser ingresada a la base de datos. Como se observa, la categorización se llega a mostrar sin las stop words, sin embargo, en la base de datos se registra el verdadero nombre de la enfermedad, en vez de “fractura rotula” se agrega “fractura de la rótula”.

En caso no encuentre una categoría por ser probable que usen algunos términos coloquiales que no están registrados en el ICD-10, entonces el diagnostico tendrá el código “ZZ100” y se le atribuirá la descripción de “no_catalogado”.

En un tratamiento posterior se deberá evaluar por qué llegó a caer en dicha categoría y registrar a la enfermedad que no llegó a calificar en la clasificación e ingresarla en el diccionario. Así cuando se reprocese dicho diagnóstico se categorizará con un código “ZZ101”. La actualización del diccionario es el último paso que podría tener el proceso y con esto culmina todo el viaje del diagnóstico desde el papel a la base datos.

CAPÍTULO VI: VERIFICACIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

6.1. Digitalización de caracteres

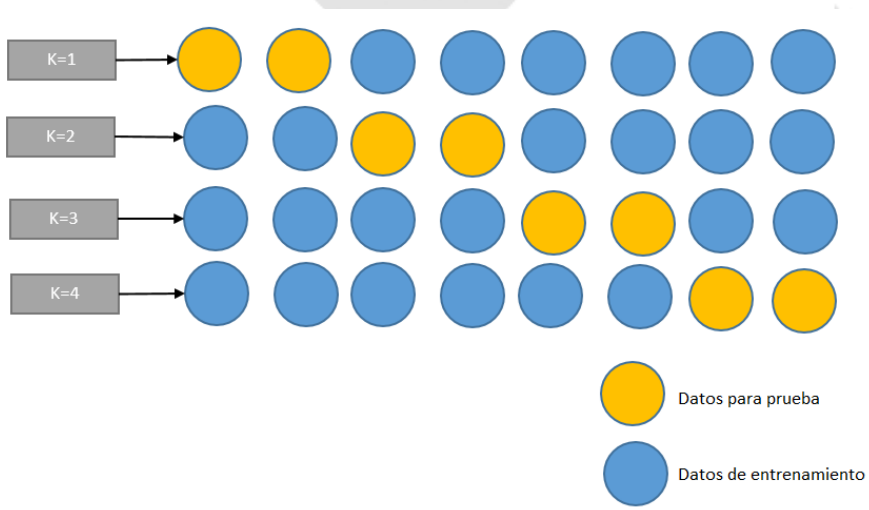
6.1.1. Tipos de validación consideradas

Esta etapa consiste en la evaluación y pruebas de la solución, para la cual se tiene planteado lo siguiente para la parte del reconocimiento de caracteres:

La técnica a aplicar será la de cross-validation, concretamente, se usará K-fold cross-validation. Como se aprecia en la figura 6.1, este consiste en separar una parte de los datos como entrenamiento y el resto se usará como pruebas de forma proporcional y consecutiva. Además, esto se repetirá K veces y en cada repetición se cogerá un diferente segmento consecutivo al anterior como entrenamiento, por lo que también se tendrán diferentes conjuntos de datos como pruebas por cada iteración. Es por ello que con esta técnica se aprovecha mejor la data que se tiene disponible puesto que cada dato es validado al menos una vez.

Figura 6.1

Cross-validation con 4 iteraciones.



Fuente: Cama Castillo (2015).

Finalmente, el resultado de cada iteración es promediada para obtener el resultado final. Como una desventaja de esto es que se vuelve más computacionalmente costoso mientras más grande sea el K y el número de datos, pero mientras más grande sea el K la estimación de errores suele ser más preciso (Grant, 2015).

El siguiente tipo de cross-validation a utilizar es el de holdout, el cual consiste en separar la data en dos grupos, el primero es usado para el entrenamiento y el otro es usado para validar el modelo (Dadhania & Dhobi, 2012). Es necesario mencionar que, estos dos conjuntos de datos son escogidos de forma aleatoria. Como beneficios de esta técnica, es más fácil de realizar cálculos, sin embargo, los datos que se tienen disponibles no son usados de la mejor manera y existe una alta varianza en los resultados (Zhu, 2005).

Adicionalmente a lo anterior, la literatura acerca del reconocimiento de caracteres plantea realizar un gold standard (Rasmussen, Luke V, et al., 2012). El cual consiste en contar y registrar cual es el verdadero valor de todas aquellas imágenes que se usen como entrenamiento, esto se realiza porque se quiere asegurar que la obtención de la precisión del clasificador sea correcta.

Una vez con todo lo anterior realizado, se podrá armar una matriz de confusión multiclase, donde cada clase representa una letra y los diferentes valores que se pudieron obtener por cada letra. Una vez con la matriz llena, se podrán obtener los valores de:

- Verdadero positivo, es el número de veces cuando el resultado obtenido coincide con el gold standard.

- Falso positivo, es el número de veces en el que el clasificador obtuvo por error una clase diferente a la que estaba en prueba.

- Falso negativo, es el número de veces cuando no se pudo detectar correctamente una clase.

- Verdadero negativo, es el número de veces donde se prueba otra clase y se obtiene esa otra que es diferente a la que estaba en prueba.

Además, se podrá obtener los siguientes valores a partir de los anteriores:

- Exactitud, este indica el ratio de resultados correctos que se pueden obtener del modelo de forma general (Old Dominion University, 2010).

- Precisión, es formulada como los verdaderos positivos sobre la suma los verdaderos positivos con los falsos positivos. En el caso de un OCR, para una letra el número total de veces en que dicha letra fue clasificada correctamente sobre el total de veces en la que se obtuvo como resultado de la clasificación a dicha letra (Hung, Luk, Yeung, Chung, & Shu, 2002).

- Sensibilidad, también conocida como exhaustividad, es formulada como los verdaderos positivos sobre la suma de los verdaderos positivos con los falsos negativos. En el caso de un OCR, el número de letras clasificadas correctamente a partir de todas las posibles letras que se podrían clasificar (Stromme & Carlson, 2010). Una alta sensibilidad significa poca probabilidad de fallar en clasificar una observación en una clase positiva.

- Especificidad, esta corresponde a la tasa en la que se obtienen verdaderos negativos o aquellas que no son miembros de una determinada clase, en otras palabra, las veces que no se obtiene una clase cuando se prueba otra clase (Ghaaliq & McCluskey, 2008). Una alta especificad implica que un clasificador rara vez falla en obtener resultados negativos, por consiguiente, una predicción positiva es un resultados altamente confiable (Math4IQB, 2013).

En la presente investigación se usará K-fold cross-validation, con un valor de K igual a 11. Si bien los valores usuales para K son 5 o 10 (Cama Castillo, 2015), se puede estimar el error de manera más precisa dado que se realizarán 11 iteraciones en un conjunto de datos de 2200 muestras por clase.

La razón del K escogido anteriormente es que se puede mantener como conjunto de datos clasificados 2000 caracteres, similar a otros caracteres, y probar con 200

caracteres por clase. Es necesario mencionar que, estos conjuntos variarán con cada iteración pues estará compuesta por un conjunto diferente de caracteres.

6.1.2. Optimización del clasificador

El objetivo de usar K-fold cross-validation es para optimizar el valor de k o el número de vecinos más cercanos que usará el clasificador (Dadhania & Dhobi, 2012). Por tanto, en cada vuelta se obtendrá los distintos niveles de precisión con los diferentes valores que pueda tomar el k del clasificador kNN.

Para realizar lo mencionado anteriormente, se empezará probando con un valor de k igual a 1 y en adelante solo se tomarán números impares (Yan Tam & Y. Kiang, 1992) en busca de aquel valor que otorgue la mejor precisión. Esto es debido a que, al aumentar el valor de k, crecerá la precisión aumenta, pero pasado un determinado valor la precisión se reduce. Por consiguiente, se escogerá el k donde la precisión sea más alta y en caso se obtenga varios valores de k con un mismo error se escogerá el menor valor para k (Dadhania & Dhobi, 2012).

Por otro lado, dado que las clases que posee el modelo son del mismo tamaño, un mayor valor de k no deberá afectar el clasificador como lo harían clases con tamaños muy diferenciadas entre sí (Dadhania & Dhobi, 2012). Además, como parte del gold standard, el conjunto de datos de prueba preparado es revisado y contado manualmente para registrar de qué letra se trata.

Resultado de lo anterior, se puede ver en el Anexo 1, se obtuvieron los siguientes valores en cuanto a la sensibilidad y tiempo para distintas cantidades de vecinos más cercanos, de 1 a 21 vecinos más cercanos. Como se puede ver en la tabla 6.1 y en la figura 6.2, claramente el valor de vecinos más cercanos a escoger será de 5, puesto que con este valor se obtiene la mayor sensibilidad de 77.5%. En cuanto al tiempo, no se aprecia una diferencia significativa entre distintos valores de k.

Es necesario mencionar que este valor obtenido de 77.5%, es una métrica referencial usada para determinar el número de vecinos a usar y no se suele ser comparada entre investigaciones. De la misma forma, el valor que se obtiene para k, puede variar

drásticamente entre distintos conjuntos de datos, como en otro estudio tomaron como un valor de k de 17 (Dadhania & Dhobi, 2012).

Tabla 6.1

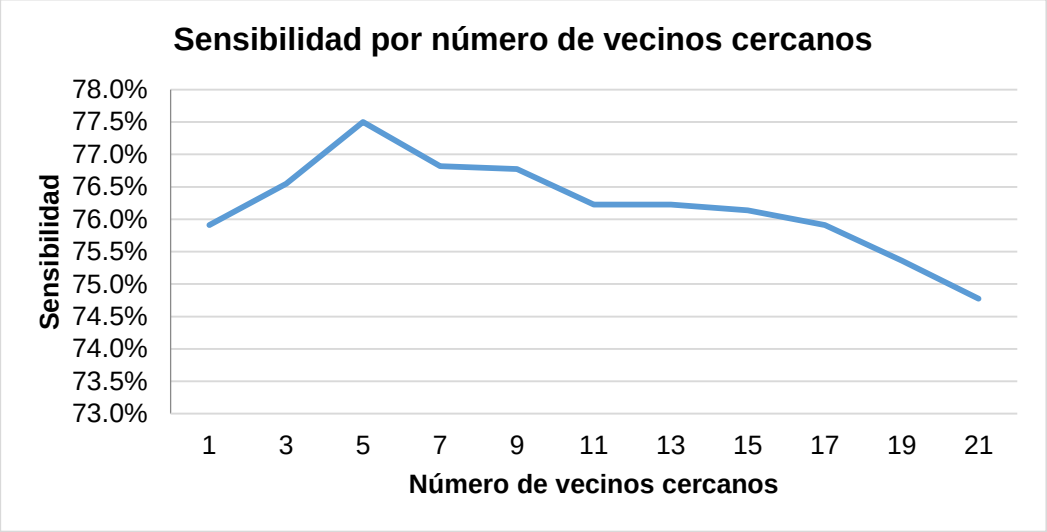
Sensibilidad y tiempo según número de vecinos cercanos obtenido de un 11-fold cross-validation

k	Sensibilidad	Tiempo (s)
1	0.759	40.936
3	0.765	35.411
5	0.775	35.152
7	0.768	34.987
9	0.768	35.188
11	0.762	34.969
13	0.762	35.493
15	0.761	37.821
17	0.759	37.835
19	0.754	37.950
21	0.748	37.770

Fuente: elaboración propia.

Figura 6.2

Sensibilidad obtenida según número de vecinos cercanos.



Fuente: elaboración propia.

6.1.3. Validación del clasificador

El segundo tipo de cross-validation a explorar es el de holdout (Biondich et al., 2002), cuyo objetivo será obtener el número de aciertos del clasificador por cada clase, es decir, por cada carácter cuantos fueron clasificados correctamente. Es por ello que, los datos se dividirán en dos grupos, los cuales explicaremos a continuación.

El primer grupo consiste en aquellas letras que fueron clasificadas, 2000 por cada clase, y el segundo consistirá en la preparación de un conjunto de datos de prueba que no hayan sido previamente clasificados conformado por 200 muestras por cada letra o clase. Por consiguiente, se usarán 104000 imágenes clasificadas y 10400 imágenes diferentes para las pruebas.

Además, como parte del gold standard, el conjunto de datos de prueba preparado es revisado y contado manualmente para asegurar de qué letra se trata. Es necesario

mencionar que, para realizar esta validación, es necesario haber obtenido el valor del k óptimo del anterior paso, esto para obtener los mejores valores por cada clase.

Resultado de esta validación, se armó una matriz de confusión multiclase, donde cada clase representa una letra en minúscula o mayúscula. Por lo que, se tienen en total 52 clases resultando en una matriz de 52x52, ver Anexo 3. A partir de esta se pudieron obtener los siguientes valores por cada clase, se manejó cada letra según su código ASCII decimal.

La exactitud que obtuvo el modelo fue de 78.69% considerando los 5 vecinos más cercanos. El valor anterior se encuentra alejado del 95.3% (Rasmussen, et al., 2012) logrado por un único clasificador como se revisó en la literatura, es por ello que se explorarán mejoras posteriormente.

En la tabla 6.2, se puede observar que la sensibilidad obtenida por clase varía entre 97.51% como máximo y un mínimo de 61.69%. Estos resultados ocurren porque existen letras cuyas minúsculas son muy parecidas a sus letras mayúsculas, esto mismo se puede ver en los resultados obtenidos de las precisiones de cada clase. En un próximo paso haremos una mejora sobre este punto.

Por otro lado, la especificidad obtenida es bastante alta por lo que se puede decir que el clasificador tiene una gran capacidad para distinguir entre clases. Este un resultado que se espera obtener, puesto que los sistemas de reconocimiento óptico deben poseer esta capacidad.

Tabla 6.2

Métricas obtenidas a partir de la matriz confusión

Clase	VP	FN	FP	VN	Sensibilidad	Precisión	Especificidad
A	171	30	24	10056	0.8507	0.8769	0.9976
B	166	35	7	10078	0.8259	0.9595	0.9993
C	132	69	71	10048	0.6567	0.6502	0.9930

(continúa)

(continuación)

D	188	13	16	10047	0.9353	0.9216	0.9984
E	183	18	3	10065	0.9104	0.9839	0.9997
F	140	61	64	10047	0.6965	0.6863	0.9937
G	183	18	3	10065	0.9104	0.9839	0.9997
H	188	13	17	10046	0.9353	0.9171	0.9983
J	162	39	44	10045	0.8060	0.7864	0.9956
K	156	45	51	10044	0.7761	0.7536	0.9949
L	195	6	50	10006	0.9701	0.7959	0.9950
M	138	63	51	10062	0.6866	0.7302	0.9950
N	193	8	47	10011	0.9602	0.8042	0.9953
O	147	54	95	10009	0.7313	0.6074	0.9906
P	132	69	64	10055	0.6567	0.6735	0.9937
Q	156	45	5	10090	0.7761	0.9689	0.9995
R	181	20	9	10061	0.9005	0.9526	0.9991
S	137	64	60	10054	0.6816	0.6954	0.9941
T	196	5	34	10021	0.9751	0.8522	0.9966
U	150	51	77	10024	0.7463	0.6608	0.9924
V	138	63	79	10034	0.6866	0.6359	0.9922
W	136	65	1	10114	0.6766	0.9927	0.9999
X	135	66	35	10081	0.6716	0.7941	0.9965
Y	150	51	71	10030	0.7463	0.6787	0.9930
Z	149	52	38	10064	0.7413	0.7968	0.9962
a	181	20	19	10051	0.9005	0.9050	0.9981
b	193	8	7	10051	0.9602	0.9650	0.9993
c	159	42	69	10023	0.7910	0.6974	0.9932

(continúa)

(continuación)

d	192	9	8	10051	0.9552	0.9600	0.9992
e	179	22	18	10054	0.8905	0.9086	0.9982
f	124	77	48	10079	0.6169	0.7209	0.9953
g	132	69	14	10105	0.6567	0.9041	0.9986
h	191	10	13	10047	0.9502	0.9363	0.9987
i	132	69	62	10057	0.6567	0.6804	0.9939
j	153	48	43	10055	0.7612	0.7806	0.9957
k	124	77	34	10093	0.6169	0.7848	0.9966
l	130	71	86	10035	0.6468	0.6019	0.9915
m	140	61	52	10059	0.6965	0.7292	0.9949
n	194	7	26	10031	0.9652	0.8818	0.9974
o	130	71	66	10055	0.6468	0.6633	0.9935
p	153	48	70	10028	0.7612	0.6861	0.9931
q	157	44	27	10067	0.7811	0.8533	0.9973
r	189	12	32	10030	0.9403	0.8552	0.9968
s	144	57	55	10052	0.7164	0.7236	0.9946
t	178	23	32	10041	0.8856	0.8476	0.9968
u	141	60	49	10061	0.7015	0.7421	0.9952
v	140	61	59	10052	0.6965	0.7035	0.9942
w	168	33	34	10049	0.8358	0.8317	0.9966
x	148	53	50	10053	0.7363	0.7475	0.9951
y	143	58	47	10061	0.7114	0.7526	0.9954
z	149	52	50	10052	0.7413	0.7487	0.9951

Fuente: elaboración propia.

6.2. Categorización de texto

En la revisión de literatura se encontraron artículos que presentaban distintas formas de validar y probar la categorización específicamente para la categorización de registros clínicos. A continuación, se desarrollará la técnica escogida: Micro-averaging (Perea, J. et al., 2008).

Las métricas utilizadas en los artículos de investigación observados se utilizarán en la presente investigación y son: Precisión (Precisión) con la que es clasificado, la exhaustividad (Recall) con la que se tuvo la clasificación y F1 - score. *(Perea, J. et al., 2009)*

- La precisión, interpretada para este caso es visto como bien categorizado por el algoritmo en una clasificación equivocada. En el caso de categorización de texto, el número total diagnósticos que fueron categorizados en su familia de enfermedad correcta sobre el total de diagnósticos que fueron clasificados en alguna familia. (Pestana R., et al., 2005)

- La exhaustividad interpretada para este caso es visto como mal categorizado por el algoritmo cuando era la clasificación correcta. En el caso de categorización de texto, el número total diagnósticos que fueron categorizados en su familia de enfermedad correcta, sobre el total de diagnósticos que eran clasificables bajo el CIE-10. (Pestana R., et al., 2005)

- El F1-score es una operación entre las dos anteriores métricas, es el doble del producto de ambas métricas sobre su suma.

Micro averaging es una técnica de validación cruzada, en esta se utilizan 10 particiones (10-fold cross-validation), en otras palabras, se repite el experimento 10 veces con distintos conjuntos de datos de entrenamiento y evaluación, así calculando, los aciertos y fallos en cada clase acumulativamente y calculando sobre estos valores. *(Perea, J. et al., 2009).* Por ejemplo:

- La fórmula del micro-averaging en dos experimentos en la métrica de precisión sería: (VP1+VP2) / (VP1+VP2+FP1+FP2)

Como se mencionó anteriormente se utilizará microaveraging y la base central del sistema metropolitano de la solidaridad (SISOL) de la municipalidad de lima que contiene diagnósticos de enfermedades dadas en dicho centro del año 2004 al 2012.

Figura 6.2

División del documento en diagnósticos

Diagnostico realizado
EL PACIENTE SE PRESENTO POR CARIES DENTAL
LA PACIENTE SUFRE DE LUMBAGO NO ESPECIFICADO
EL PACIENTE INGRESA DEBIDO A TRASTORNOS DE LA ACOMODACION Y DE LA REFRACCION
EL PACIENTE SE PRESENTO POR FARINGITIS AGUDA, NO ESPECIFICADA
EL PACIENTE ENCUENTRA SINTOMAS DE PADECER DISPEPSIA
EL PACIENTE ENCUENTRA SINTOMAS DE PADECER RINOFARINGITIS AGUDA [RESFRIADO COMUN]
EL PACIENTE INGRESA DEBIDO A HIPERTENSION ESENCIAL (PRIMARIA)
EL PACIENTE ENCUENTRA SINTOMAS DE PADECER INFECCION DE VIAS URINARIAS, SITIO NO ESPECIFICADO
LAS CAUSAS MEDICAS DE SU INGRESO SON DIARREA Y GASTROENTERITIS DE PRESUNTO ORIGEN INFECCIOSO
LA PACIENTE SUFRE DE NECROSIS DE LA PULPA
EL PACIENTE PRESENTA SINTOMAS DE GASTRITIS Y DUODENITIS

Fuente: elaboración propia

Para dar contexto a los diagnósticos del SISOL se decidió adjuntar un pequeño texto adelante aleatoriamente en base a lo utilizado en el manual de codificación CIE-10 del ministerio de sanidad, servicios sociales e igualdad de España. La validación cruzada se hará con 55 diagnósticos cada vuelta.

Así mismo también se utiliza el mismo corpus de la ICD para tomar como características palabras claves de la familia de la enfermedad y también como datos de entrenamiento.

Figura 6.3

Familia y subcategoría de enfermedades de la ICD

idFamilia	Descripcion familia	idEnfermedad	descripcionEnfermedad
M47	ESPONDILOSIS	M478	OTRAS ESPONDILOSIS
M47	ESPONDILOSIS	M479	ESPONDILOSIS, NO ESPECIFICADA
M48	OTRAS ESPONDILOPATIAS	M480	ESTENOSIS ESPINAL
M48	OTRAS ESPONDILOPATIAS	M481	HIPEROSTOSIS ANQUILOSANTE [FORESTIER]
M48	OTRAS ESPONDILOPATIAS	M482	ESPONDILOPATIA INTERESPINOSA (VERTEBRAS EN BESO)
M48	OTRAS ESPONDILOPATIAS	M483	ESPONDILOPATIA TRAUMATICA

Fuente: elaboración propia.

Con todo ello se realizará las pruebas hasta el nivel de enfermedad, primando la validez de la familia principal (3 primeros caracteres).

Figura 6.4 Familia y subcategorías de la ICD

```
Inicio:  4617.0
Paso la prueba:  LA PACIENTE INGRESA DEBIDO A INFECCIONES HERPETICAS [HERPES SIMPLE] :  P582  ICTERICIA NEONATAL D
Paso la prueba:  EL PACIENTE PRESENTA OBESIDAD :  E661  OBESIDAD:  OBESIDAD INDUCIDA POR DROGAS
Paso la prueba:  LA PACIENTE SE PRESENTA POR OTRAS INFECCIONES AGUDAS DE SITIOS MULTIPLES DE LAS VIAS RESPIRATORIA
Paso la prueba:  EL PACIENTE PRESENTA SINTOMAS DE ACNE VULGAR :  F312  TRASTORNO AFECTANIVO BIPOLAR:  TRASTORNO AF
Paso la prueba:  EL PACIENTE TIENE ESGUINCES Y TORCEDURAS DEL TOBILLO :  S034  LUXACION ESGUINCE Y TORCEDURA DE AR
Paso la prueba:  INGRESA POR (OSTEO)ARTROSIS PRIMARIA GENERALIZADA :  G71  TRASTORNOS MUSCULARES PRIMARIOS:  TRAST
Paso la prueba:  LA PACIENTE SUFRE DE TRASTORNO DE ANSIEDAD GENERALIZADA :  F413  OTROS TRASTORNOS DE ANSIEDAD:  O
Paso la prueba:  LA PACIENTE PRESENTA ANEMIA POR DEFICIENCIA DE HIERRO SIN OTRA ESPECIFICACION :  B19  HEPATITIS V
Final:  4650.0
```

Fuente: elaboración propia

Tabla 6.3

Familia y subcategorías de la ICD

REAL	CATEGORIZADO	NRO
A01	A01	1
A63	A63	1
B01	B01	1
B19	B19	1
B37	B37	1
D17	D36	1
E05	E05	1

Fuente: elaboración propia.

En la tabla 6.4 se puede ver un ejemplo de la familia real, así como la categorizada en base a los diagnósticos ingresados. La columna “Real” indicada la familia que pertenecía al diagnóstico, mientras la columna “categorizado” pertenece al resultado del sistema.

Llevando acabo el microaveraging se tomarían las medidas en 10 pruebas (distribuidas en el gráfico en 10 columnas con sus respectivos números), cada uno de 55 diagnósticos tipeados a mano. A partir de estos, la información se ingresará al sistema y se evaluará las métricas propuestas. En el siguiente cuadro se muestran los resultados:

Tabla 6.4

Resultados de las métricas

	1	2	3	4	5	6	7	8	9	10	MICRO
VP	44	43	42	44	41	41	43	49	43	42	432
FP	3	4	2	4	4	8	5	3	2	5	40
VN	0	2	0	3	0	1	0	0	1	1	8
FN	8	6	11	4	10	5	7	3	9	7	70
PR	93.62%	91.49%	95.45%	91.67%	91.11%	83.67%	89.58%	94.23%	95.56%	89.36%	91.53%
EX	84.62%	87.76%	79.25%	91.67%	80.39%	89.13%	86.00%	94.23%	82.69%	85.71%	86.06%
F1-SC	88.89%	89.58%	86.60%	91.67%	85.42%	86.32%	87.76%	94.23%	88.66%	87.50%	88.71%

Fuente: elaboración propia.

La métrica final y con la que se evaluará será el F1-Score que básicamente con su fórmula involucra a la precisión y exhaustividad. Además, también se evalúa tanto la precisión como exhaustividad individualmente para ver oportunidades de mejora respectivas del modelo y las medidas a tomar.

En esta ocasión el mínimo valor que tomo la precisión fue de 83.67%, mientras que la exhaustividad fue de 79.25%. El menor valor del F1-Score fue de 85.42% que representa un porcentaje alentador a pesar que la cifra se sitúe por debajo del 90%. Esto se debe a que, en ocasiones al haber algunas categorías iguales de enfermedades, el software clasificaba en una categoría “G80” cuando era “G81” como se observa en la tabla 6.4.

El resultado final bajo el esquema de MicroAveraging es una precisión del 91.53%, una exhaustividad del 86.06% y un F1-Score de 88.71%. Así como en los

estudios anteriores explorados en la revisión de literatura, a comparación de otros algoritmos con un F1-score rondando en promedio los 85% la técnica evidencia ser un eficaz clasificador que logra catalogar multiclases en el contexto de categorización de enfermedades.

6.3. Mejoras de la solución

En cuanto a la digitalización del texto, producto de estos resultados que no son tan buenos en su mayoría se decidió por aplicar una técnica de enderezado sobre los datos de esta forma resultan más normalizados de esta forma se pueden obtener mejores resultados (LeCun, Cortes, & Burges, 1998). Un ejemplo de la aplicación de esta técnica se puede ver en la figura 6.6.

Figura 6.5

Imágenes sin enderezar (izquierda), imágenes enderezadas (derecha)

Fuente: elaboración propia

Debido a que se volvió a clasificar las letras enderezándolas y se realizaron las pruebas anteriormente mencionadas, ver Anexo 2. Primero se tuvo que obtener el valor de k óptimo mediante K-fold cross-validation y posteriormente realizar holdout cross-validation para volver a realizar una nueva matriz confusión. Con ello se obtuvieron los resultados que se ven en la tabla 6.5 y la figura 53.

Tabla 6.5

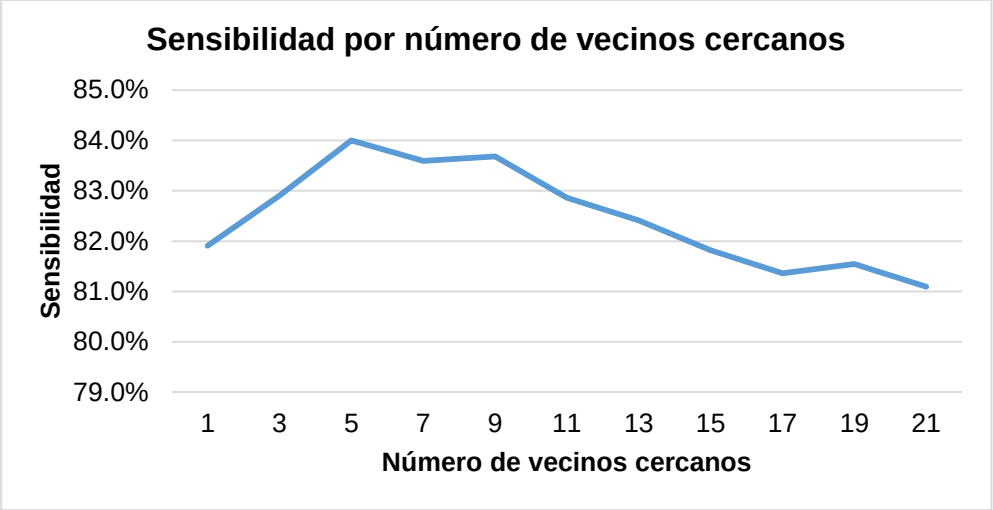
Sensibilidad y tiempo según número de vecinos cercanos obtenido de un 11-fold cross-validation con letras enderezadas

k	Sensibilidad	Tiempo (s)
1	0.8191	46.563
3	0.8291	40.296
5	0.8400	39.698
7	0.8359	39.227
9	0.8368	39.644
11	0.8286	39.571
13	0.8241	39.602
15	0.8182	38.531
17	0.8136	38.740
19	0.8155	38.308
21	0.8109	38.802

Fuente: elaboración propia.

Figura 6.7

Sensibilidad obtenida con 11-fold cross-validation con letras enderezadas.



Fuente: elaboración propia.

Para este nuevo conjunto de datos normalizado a partir de la aplicación de K-fold cross-validation, se obtuvo que el número de vecinos más cercanos óptimo es de 5, como se ve en la figura 6.7. El valor obtenido es el mismo al obtenido anteriormente, sin embargo, la mayor diferencia es la exactitud lograda puesto que de obtener un 77% pasó a obtener un 84%.

Por otro lado, como se ve en la tabla 6.5, el tiempo de procesado aumento en un aproximado de 3 a 4 segundos, esto ocurre porque la normalización es una capa más que se adiciona a todo el preprocesamiento planteado anteriormente. Una vez con este valor, se prosigue a realizar una matriz de confusión, Anexo 4. De forma similar a la anterior, se armó la tabla 6.6, en la que se tuvo una exactitud del modelo de 79.53% la cual no resultó ser un incremento significativo.

Tabla 6.6

Métricas obtenidas a partir de la matriz confusión con letras enderezadas

Clase	VP	FN	FP	VN	Sensibilidad	Precisión	Especificidad
A	180	21	18	10053	0.8955	0.9091	0.9982
B	174	27	2	10075	0.8657	0.9886	0.9998

(continúa)

(continuación)

C	**143**	**58**	**77**	**10031**	**0.7114**	**0.6500**	**0.9924**
D	185	16	14	10052	0.9204	0.9296	0.9986
E	185	16	1	10065	0.9204	0.9946	0.9999
F	133	68	51	10067	0.6617	0.7228	0.9950
G	183	18	1	10067	0.9104	0.9946	0.9999
H	185	16	16	10050	0.9204	0.9204	0.9984
I	165	36	157	9929	0.8209	0.5124	0.9844
J	153	48	34	10064	0.7612	0.8182	0.9966
K	163	38	57	10031	0.8109	0.7409	0.9943
L	195	6	26	10030	0.9701	0.8824	0.9974
M	140	61	64	10047	0.6965	0.6863	0.9937
N	193	8	25	10033	0.9602	0.8853	0.9975
O	142	59	88	10021	0.7065	0.6174	0.9913
P	128	73	41	10082	0.6368	0.7574	0.9959
Q	166	35	6	10079	0.8259	0.9651	0.9994
R	182	19	4	10065	0.9055	0.9785	0.9996
S	128	73	72	10051	0.6368	0.6400	0.9929
T	197	4	30	10024	0.9801	0.8678	0.9970
U	144	57	61	10046	0.7164	0.7024	0.9940
V	150	51	79	10022	0.7463	0.6550	0.9922
W	125	76	19	10107	0.6219	0.8681	0.9981
X	143	58	37	10071	0.7114	0.7944	0.9963
Y	155	46	46	10050	0.7711	0.7711	0.9954
Z	153	48	31	10067	0.7612	0.8315	0.9969
a	188	13	22	10041	0.9353	0.8952	0.9978

(continúa)

(continuación)

b	**193**	**8**	**4**	**10054**	**0.9602**	**0.9797**	**0.9996**
c	152	49	58	10041	0.7562	0.7238	0.9943
d	192	9	11	10048	0.9552	0.9458	0.9989
e	177	24	23	10051	0.8806	0.8850	0.9977
f	134	67	60	10057	0.6667	0.6907	0.9941
g	141	60	14	10096	0.7015	0.9097	0.9986
h	196	5	20	10035	0.9751	0.9074	0.9980
i	126	75	49	10076	0.6269	0.7200	0.9952
j	150	51	38	10063	0.7463	0.7979	0.9962
k	131	70	31	10089	0.6517	0.8086	0.9969
l	122	79	63	10066	0.6070	0.6595	0.9938
m	129	72	47	10075	0.6418	0.7330	0.9954
n	196	5	28	10027	0.9751	0.8750	0.9972
o	135	66	65	10051	0.6716	0.6750	0.9936
p	158	43	75	10018	0.7861	0.6781	0.9926
q	175	26	34	10042	0.8706	0.8373	0.9966
r	191	10	33	10027	0.9502	0.8527	0.9967
s	132	69	59	10060	0.6567	0.6911	0.9942
t	187	14	26	10038	0.9303	0.8779	0.9974
u	137	64	60	10054	0.6816	0.6954	0.9941
v	144	57	53	10054	0.7164	0.7310	0.9948
w	172	29	55	10024	0.8557	0.7577	0.9945
x	153	48	57	10041	0.7612	0.7286	0.9944
y	149	52	48	10054	0.7413	0.7563	0.9952
z	162	39	50	10039	0.8060	0.7642	0.9950

Fuente: elaboración propia.

En cuanto a la sensibilidad, se puede apreciar que existen algunas clases que muestran mejoras entre un 4 a 5% mientras que otras clases disminuyen en un 5% pero en su gran mayoría representan una mejora.

Además, a partir de la matriz de confusión se pudo obtener que las letras mayúsculas son confundidas por sus minúsculas, esto sucede especialmente con las letras ‘C’, ‘F’,’J’,’K,’M’,’O’,’P’,’S’,’U’,’V ’,’W ’,’X ’,’Y ’,’Z’. Así mismo, las siguientes las letras en minúsculas son confundidas con sus mayúsculas correspondientes ‘c’, ‘i’, ’j’, ’m ’, ’o ’, ’p ’, ’s ’, ’t ’, ’u ’, ’v ’, ’w ’, ’x ’, ’y ’, ’z’.

Estos casos donde las letras son confundidas con su mayúscula o minúscula representan la gran mayoría de errores en el modelo lo que ocasionan una gran disminución en su exactitud. Puesto que, los resultados obtenidos son considerando la obtención de mayúsculas por minúsculas o viceversa como errores.

Adicionalmente, existen casos especiales donde la letra ‘h’ es confundida en menor medida con la ‘t’, ‘q’ con ‘g’ , ‘x’ con ‘Y’ e ‘y’ con ‘v’ y en otros casos que suceden con mayor frecuencia la ‘i’ con ‘l’, ‘Q’ con ‘O’ y ‘a’ ,’e’ con ‘c’, ‘r’ con ‘g’, ‘j’ con ‘i’. Lo anterior sucede porque las letras que fueron usadas como clasificación son muy similares a las letras en prueba.

Adicionalmente, se puede apreciar que en ambas matrices de confusión existe una gran parte de los resultados que fueron correctamente clasificados. Existe también, una segunda porción de datos significativa en tamaño que no fue correctamente clasificada, los cuales corresponden a errores en los que se obtuvieron la letra en minúscula o viceversa.

Debido a que, en próximos pasos de la investigación, las letras serán convertidas a minúsculas se puede dejar de considerar el evento mencionado anteriormente como un error. Por lo que surge una nueva matriz de confusión, ver Anexo 5, con ello los resultados se resumen en la tabla 6.7.

Tabla 6.7

Métricas obtenidas a partir de la matriz confusión con letras enderezadas sin considerar como error letras mayúsculas y minúsculas.

Clase	VP	FN	FP	VN	Sensibilidad	Precisión	Especificidad
A	180	21	18	10063	0.8955	0.9091	0.9982
B	174	27	2	10085	0.8657	0.9886	0.9998
C	194	7	35	10032	0.9652	0.8472	0.9965
D	185	16	14	10062	0.9204	0.9296	0.9986
E	188	13	1	10072	0.9353	0.9947	0.9999
F	187	14	7	10067	0.9303	0.9639	0.9993
G	183	18	1	10077	0.9104	0.9946	0.9999
H	187	14	16	10058	0.9303	0.9212	0.9984
I	170	31	111	9980	0.8458	0.6050	0.9890
J	184	17	16	10061	0.9154	0.9200	0.9984
K	193	8	3	10065	0.9602	0.9847	0.9997
L	196	5	26	10039	0.9751	0.8829	0.9974
M	187	14	3	10071	0.9303	0.9842	0.9997
N	194	7	25	10042	0.9652	0.8858	0.9975
O	195	6	30	10036	0.9701	0.8667	0.9970
P	193	8	10	10058	0.9602	0.9507	0.9990
Q	167	34	6	10088	0.8308	0.9653	0.9994
R	182	19	3	10076	0.9055	0.9838	0.9997
S	184	17	9	10068	0.9154	0.9534	0.9991
T	200	1	21	10040	0.9950	0.9050	0.9979

(continúa)

(continuación)

U	192	9	12	10057	0.9552	0.9412	0.9988
V	193	8	30	10038	0.9602	0.8655	0.9970
W	178	23	2	10081	0.8856	0.9889	0.9998
X	192	9	3	10066	0.9552	0.9846	0.9997
Y	193	8	14	10054	0.9602	0.9324	0.9986
Z	198	3	5	10058	0.9851	0.9754	0.9995
a	188	13	22	10051	0.9353	0.8952	0.9978
b	193	8	4	10064	0.9602	0.9797	0.9996
c	194	7	7	10060	0.9652	0.9652	0.9993
d	192	9	11	10058	0.9552	0.9458	0.9989
e	177	24	20	10064	0.8806	0.8985	0.9980
f	178	23	6	10077	0.8856	0.9674	0.9994
g	141	60	14	10106	0.7015	0.9097	0.9986
h	196	5	18	10047	0.9751	0.9159	0.9982
i	172	29	44	10045	0.8557	0.7963	0.9956
j	168	33	7	10086	0.8358	0.9600	0.9993
k	185	16	1	10075	0.9204	0.9946	0.9999
l	122	79	62	10077	0.6070	0.6630	0.9939
m	190	11	0	10071	0.9453	1.0000	1.0000
n	196	5	27	10038	0.9751	0.8789	0.9973
o	193	8	12	10056	0.9602	0.9415	0.9988
p	189	12	10	10062	0.9403	0.9497	0.9990
q	175	26	33	10053	0.8706	0.8413	0.9967
r	192	9	33	10036	0.9552	0.8533	0.9967
s	195	6	3	10063	0.9701	0.9848	0.9997

(continúa)

(continuación)

t	196	5	23	10042	0.9751	0.8950	0.9977
u	196	15	12	10043	0.9289	0.9423	0.9988
v	193	8	10	10058	0.9602	0.9507	0.9990
w	189	12	2	10070	0.9403	0.9895	0.9998
x	187	14	8	10066	0.9303	0.9590	0.9992
y	181	20	10	10070	0.9005	0.9476	0.9990
z	188	13	5	10068	0.9353	0.9741	0.9995

Fuente: elaboración propia.

A partir de los valores anteriores, se puede concluir que una precisión en aproximado de 92% es bastante alta, lo que significa que el clasificador categoriza de forma incorrecta un 8% de las veces cuando se encuentra frente a una determinada clase. Un comportamiento similar ocurre con la sensibilidad, donde el modelo retorna la clase seleccionada un 92% de las veces y no otras clases.

Sin embargo, existen excepciones a lo anterior porque en los resultados obtenidos se pueden apreciar clases que resultaron con estas métricas con valores muy diferentes a las demás. Uno de estos casos es de la ‘I’ (‘i’ mayúscula) que posee una precisión baja lo que significa que muy frecuentemente se obtuvo esta clase por error cuando se estaba evaluando otras clases esto es debido a su gran similitud con la letra ‘l’ (‘L’ minúscula) y la misma ‘i’.

De forma similar a la anterior, una baja precisión sucede también con la letra ‘l’ (‘L’ minúscula), la cual también presenta una baja sensibilidad, esto quiere decir que se obtuvieron otras letras cuando se estaba probando esta letra. Así mismo, se puede apreciar que en el caso de ‘g’ se aprecia una baja sensibilidad.

Por otro lado, al ver la especificidad permite concluir que el clasificador puede diferenciar entre clases un promedio de 99.84% de veces, por lo que son pocos los casos en los que no se obtiene una determinada clase por error cuando se prueba otra clase. Esto

significa que los resultados son confiables. Este resultado es ligeramente mayor a la especificidad revisada en la literatura de 99.6% (Rasmussen, et al., 2012).

Finalmente, la exactitud del modelo conseguido aumenta un 13.41% respecto a nuestra prueba inicial, esto significa que se obtuvieron las letras correctas un 92.1% de las veces en las que se probó. Este es un resultado bastante comparable con los 95.3% usando un solo clasificador (Rasmussen, et al., 2012) en la literatura revisada.

La diferencia anteriormente mencionada, radica en que en la literatura se usaron distintos programas de uso comercial y que su sistema se entrenó con un número limitado de caracteres, restricción que no podemos realizar. Por otro lado un resultado más comparable visto en la literatura es 92.4% (Biondich et al., 2002) donde usan un software comercial y se limita a la digitalización de dígitos.

Por otro lado, en cuanto a la categorización del texto, una de las mejoras fundamentales es dar un trabajo más profundo en el procesamiento del lenguaje natural (NLP). Se probaron hasta 3 “taggers” en la realización del presente trabajo de investigación: Spaghetti-tagger, StanfordPOSTagger y Treetagger.

Figura 6.6

Spaghetti-Tagger

```
[('Osteoporosis', None), ('relacionada', None), ('con', u'sps00'), ('la',
u'da0fs0'), ('edad', u'ncfs000'), ('y', u'cc'), ('leve', u'aq0cs0'),
('fractura', u'ncfs000'), ('en', u'sps00'), ('la', u'da0fs0'), ('rotula,',
None), ('reingresante.', None)]
```

Fuente: elaboración propia

Después de las pruebas el paquete de StanfordPOSTagger en español fue el seleccionado para desempeñar la función y estructurar las oraciones de los diagnósticos. Sin embargo, aún había algunos mínimos detalles en el contexto, pues catalogaba algunos sustantivos como verbos, adjetivos como adverbios, etc. De ser solucionados, se considera que potenciaría más el clasificador a la hora de extraer las posibles categorías.

Adicionalmente, se realizaron pruebas con letras escritas a mano que no se encontraban en la base de datos. Para ello, se procedió a escribir un pequeño texto y

proceder a su escaneo para luego ejecutarlo en la aplicación desarrollada, durante esta prueba se presentaron dificultades.

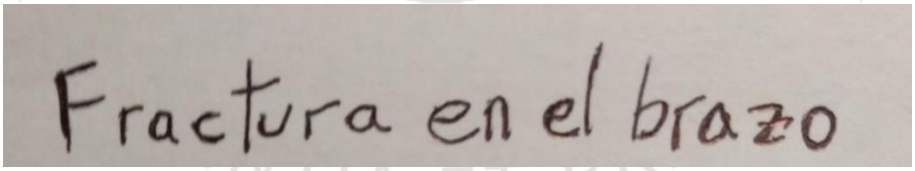
La primera dificultad presentada fue una resolución del texto escaneado diferente a la esperada, cuando es muy alta influye en el número de pixeles vecinos para realizar la umbralización por lo que puede afectar en el tiempo de procesamiento. Así mismo afectó, el número de iteraciones utilizadas en la dilatación para juntar las letras en palabras. Como solución rápida en los casos anteriores se tuvieron que ajustar dichos parámetros.

Otra dificultad presentada fue el reflejo de la tinta con la luz, esto hace parecer que las letras presentan huecos, cuando en verdad no los tienen, como se ve en la figura 6.9 en la ‘z’. Adicionalmente, el papel al ser escaneado se podría ver notablemente las fibras del papel por lo que todo esto era recogido como ruido durante el preprocesamiento.

Para superar este último problema, se disminuyó el contraste y luego de preprocesar se obtuvo una imagen con muchísimo menos ruido en la figura 6.10. Con lo anterior, las letras presentan menos agujeros y se pudo obtener el resultado en la figura 6.11, donde el clasificador solo llegó a clasificar incorrectamente las ‘r’.

Figura 6.7

Imagen original con ruido presente en la letra “z”.



Fuente: elaboración propia

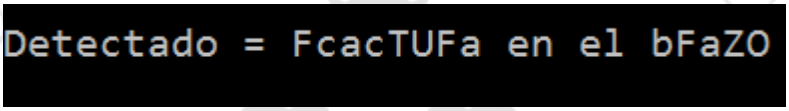
Figura 6.10

Imagen preprocesada con ruido eliminado.

Fuente: elaboración propia

Figura 6.11

Resultado de procesamiento de una frase.



Fuente: elaboración propia

6.4. Discusión

Para la parte de reconocimiento de caracteres se presentaron problemas en el reconocimiento puesto que inicialmente se obtuvo una exactitud 77%. La anterior tenía que ser mejorada, por lo que se aplicó una técnica de normalización con lo que se pudo mejorar los resultados. Sin embargo, no fue un cambio significativo, en este punto se decidió que no considerar como error que se obtengan letras mayúsculas por minúsculas o viceversa el resultado obtenido es mayor a 92%.

Durante la investigación también se pudo aplicar una técnica de adelgazamiento, pero esta no fue suficientemente buena ya que se obtuvieron resultados mediocres. Si bien la tasa obtenida es alta el modelo suele fallar en determinadas letras las cuales son muy parecidas entre sí, todos los casos encontrados fueron mencionados anteriormente.

Al tratar de reconocer textos escritos a mano que no se encontraban en la NIST SD19, permitió encontrar factores que no se consideraron previamente como la tinta del lapicero, el papel e inclusive la iluminación. No se tomaron en cuenta estos factores

puesto que la base de datos te otorga las letras casi preprocesadas, razón por la cual se tuvo que realizar algunas modificaciones respecto a parámetros usados en la solución

Por otra parte, el pre procesamiento de la información una vez digitalizada fue una de las fases más difíciles debido a que en español no se ha desarrollado tantas soluciones, los corpus que aparecían estaban en inglés y sin mencionar las estructuras gramaticales. Sin duda lo más desafiante ha sido poder trabajar con esta dificultad de no encontrar corpus que pudieran aportar efectivamente al presente trabajo de investigación.

Adicionalmente, la plataforma (Python 2.7) presentaba ciertos problemas con el encoding, por lo que las palabras que se ofrecían en "español" no siempre eran correctas o había ciertos reemplazos en palabras con tildes, por lo que se optó limpiar las tildes y "ñ" para un mejor procesamiento.

Además, en la literatura se sugería utilizar la técnica SVM, sin embargo, el procedimiento de cómo y qué parámetros se utilizaban no estaban descritos. Esta fue una dificultad encontrada durante el desarrollo del software puesto que se tuvo que iniciar una investigación más profunda respecto al tema. Igualmente, con la técnica explorada (TF-IDF) y su enlace a la técnica SVM, ha resultado desafiante encontrar el procedimiento pues nuevamente la literatura solo mencionaba la efectividad y los resultados, más no el proceso.

Por último, al categorizar los diagnósticos médicos se pudo encontrar dificultades también al utilizar los datos de entrenamiento, pues igualmente no se contaba con un corpus ya definido con las respuestas por lo que la definición de las categorías correctas se hizo manualmente con diagnósticos médicos del SISOL y el manual del ICD-10 dado por el ministerio español. Tanto la revisión como validación fueron hechas uno a uno, pues no se encontraban identificadas con su par en la base ICD-10.

CAPÍTULO VII: CONCLUSIONES

7.1. Conclusiones

- El presente estudio es una buena base para la clasificación de términos médicos con letra imprenta partiendo de su digitalización hasta la categorización de enfermedades en base al ICD-10. Esto fue posible a que se pudo identificar un clasificador de letras escritas a mano que permitiera obtener buenos resultados digitalizando diagnósticos médicos utilizando el NIST Special Database 19.

- Según las etapas seleccionadas, el preprocesamiento escogido fue adecuado según el clasificador seleccionado para el reconocimiento de los caracteres de la base de datos, sin embargo, los parámetros usados para este tienen que ser cambiados para obtener una mejor exactitud. Además, se concluye que el reconocimiento de caracteres tiene buen funcionamiento por lo que el clasificador kNN demuestra un gran potencial, sin embargo, como fue mencionado anteriormente tiene excepciones con aquellas letras que son muy parecidas entre sí, esto se puede superar usando un corrector ortográfico o una solución más compleja.

- Además, el preprocesamiento mediante las técnicas de procesamiento de lenguaje natural ha sido desafiante y es mejorable con trabajo más especializado como se mencionó anteriormente. En suma, el algoritmo Support vector machine ha demostrado en la literatura grandes resultados que han podido ser igualados en la solución.

- Finalmente, el motivo de esta primera parte es encontrar un clasificador que permita una buena tasa de aciertos, si bien para algunos caracteres este no es el caso, el sistema en general ayuda a cumplir este objetivo al tener tasas altas de reconocimiento de caracteres. Así mismo, en complemento la segunda parte permite categorizar los términos a la familia de enfermedades a primer nivel más

cercana a la que esté, con una técnica de aprendizaje de maquina no tan detallada en los trabajos de investigación revisados.

CAPÍTULO VIII: RECOMENDACIONES Y TRABAJOS FUTUROS

8.1. Recomendaciones

Si bien se pudieron mejorar en gran medida los resultados obtenidos, todavía existe espacio para mejoras por ejemplo, realizar una extracción de características (Sun, 2015), durante el estudio solo se probó con adelgazamiento de la imagen. Sin embargo, los resultados obtenidos no resultaron como los esperados puesto que se obtuvieron resultados mediocres con entre 30% a 40% de sensibilidad, ver Anexo 6 y 7, lo cual representa menos de la mitad de lo que se pudo obtener sin aplicar esta técnica.

Por otro lado, al combinar esta técnica con un enderezado como se ve en la figura 8.1, se pudo mejorar los resultados en un 4% en algunos casos. Esto fue debido a que no solamente se tiene que aplicar esta técnica sola, sino se tienen que aplicar varias técnicas para extraer múltiples características para obtener mejores resultados (Ouchtati, Redjimi, & Bedda, 2015).

Figura 8.1 de la imagen

Imagen enderezada (Izquierda), imagen adelgazada (derecha).

Fuente: elaboración propia

Para la corrección de ciertos caracteres que fueron clasificados incorrectamente se podría utilizar un corrector basado en el contexto en el que se encuentran las palabras o por otro lado una persona encargada podría procesar estos errores. Esta etapa ya sería considerada como parte de un preprocesamiento (Bushinak, AbdelGaber, & AlSharif, 2011).

Respecto a la categorización de textos se recomienda realizar un estudio más profundo en las técnicas de procesamiento de lenguaje natural que no fueron tratadas y afinar mejor la categorización de enfermedades, esto es un chunking mejorado y un mejor POStagging para encontrar más preciso el rol de la palabra, así como un trabajo más detallado en el contexto.

8.2. Trabajo futuros

Evidentemente toda la investigación ha presentado algunos supuestos y los resultados que demuestran poder ser raíz o fuente de trabajos futuros, punto que se explicará a continuación.

Este trabajo puede presentar gran potencial considerando letra corrida y teniendo en cuenta las palabras enteras y letras conectadas entre sí para realizar el reconocimiento. Para lo que puede ser necesario agregar técnicas de preprocesamiento, como proyección de histograma, frecuentemente usada. Finalmente, una variable crítica de un sistema de este tipo será la legibilidad en la letra del doctor.

Una de las mejoras a considerar es que el clasificador de caracteres maneje los propios del alfabeto castellano como es la “ñ” e inclusive llegar a considerar acentos en las vocales, lo que implicaría también considerar un corpus en español. Esto se dejó de lado puesto que la base de datos usada durante el estudio solo tomo en consideración las letras del habla inglesa. Sin embargo, esto puede ser fácilmente corregido en la etapa de postprocesamiento con un autocorrector ortográfico.

Otro punto importante es respecto a la contextualización. La digitalización del texto no es precisa al 100% por lo que es necesario este enlace para limpiar lo que se recibe de dicho proceso. Para evaluar el contexto de las palabras, como se vio en las pruebas extras, se digitalizaron 3 letras erróneamente lo que causaría, sin este preprocesamiento, confusión y con ello una mala categorización.

El presente trabajo no pudo ser probado en un entorno real como lo sería un consultorio médico, por lo que una vez superado el reconocimiento de letra corrida sería interesante probar la idea para ver el efecto que se tiene sobre el doctor y su entorno de trabajo.

Al considerar que la solución se ejecute en tiempo real y puesto que se apostó por un reconocimiento offline, no se profundizó en optimizaciones de memoria y tiempo de ejecución. Por lo expuesto anteriormente, un punto de mejora puede ser integrarlo con otras herramientas como tabletas o lapiceros digitales que permitan reconocer los caracteres mientras son escritos.

Finalmente, la presente solución al tener los distintos diagnósticos clasificados y ya en una base de datos podría ser usada como fuente para futuras investigaciones que quieran aprovechar de registros médicos por ejemplo para predecir enfermedades, encontrar tendencias dentro de esta, entre muchos otros ejemplos que podrían surgir.

GLOSARIO

A continuación, definimos los términos más importantes de la investigación:

1. **Cross-validation:** es una técnica usada para la validación de un modelo, la cual consiste en separar una parte de los datos como entrenamiento y el resto se usará como pruebas de forma proporcional y consecutiva. Donde se tendrán diferentes conjuntos de datos como pruebas por cada iteración por lo que se aprovecha al máximo la data disponible.
2. **Ruido impulsional:** este es el tipo de ruido más común al digitalizar una imagen. Se caracteriza al tener
3. **k Nearest Neighbor (kNN):** es un método estadístico que permite asignar o clasificar nuevos elementos a un conjunto de datos que tienen un mayor parecido a una serie de parámetros acerca de un conjunto de datos.
4. **National Institute of Satandards and Technology Special Database 19 (NIST SD19):** es una base de datos que contiene 814255 caracteres escritos a mano por 4169 personas. En esta se pueden encontrar letras en mayúscula, minúscula y dígitos.
5. **UMLS:** Es un sistema que contiene gran cantidad de información útil como vocabularios o clasificaciones para la clasificación de texto en el campo biomédico.
6. **Tokenización:** Es el proceso por el cual se divide al texto en distintas unidades llamadas token.
7. **Wordnet:** Es una base de datos léxica disponible en ingles que incluye sinónimos, algunas definiciones y distintas relaciones entre estos sinónimos.
8. **Metatesauro:** Es parte de UMLS, es como un gran diccionario donde están los conceptos o significados de diversos vocabularios.
9. **ICD-10:** Es el International Statistical Classification of Diseases and Related Health Problems versión 10 cuyas siglas son ICD-10, proveen un catálogo de enfermedades o problemas de la salud.
10. **SVM:** Es una técnica de aprendizaje de máquina que clasifica en base a la representación de los datos en el espacio y el uso de un hiperplano para dividir los puntos y así predecir a los espacios al que pertenecen los nuevos datos.

11. **POSTagger:** POS significa Part of Speech, por lo que POS Tagger, indica la clasificación por parte de oración, por ejemplo, sustantivo, verbo, etc. En la presente solución se utiliza el StanfordPOSTagger que está en español.
12. **Chunking:** Es el proceso por el cual se escoge ciertos patrones de estructuras morfológicas (sustantivo-verbo-sustantivo, por ejemplo) que resultan relevantes para el análisis de texto.
13. **Chiking:** Es el proceso en el cual se elimina las estructuras morfológicas innecesarias para la clasificación.
14. **Stop words:** Es un término utilizado en el procesamiento de lenguaje natural para nombrar a aquellas palabras que no aportan al análisis del texto, suelen ser palabras comunes como preposiciones.
15. **NLTK:** Es una librería que cuenta con distintas funciones para el procesamiento del lenguaje natural.
16. **Microaveraging:** Es una técnica de validación cruzada, en esta se utilizan 10 particiones (10-fold cross-validation), en otras palabras, se repite el experimento 10 veces con distintos conjuntos de datos de entrenamiento y evaluación, así calculando, los aciertos y fallos en cada clase acumulativamente y calculando sobre estos valores.
17. **F1-Score:** Es una métrica que resulta de la operación entre la precisión y exhaustividad, es el doble del producto de ambas métricas sobre su suma.

REFERENCIAS

Abecassis, F. (Octubre, 2011). *OpenCV - Rotation (Deskewing)*. Recuperado de: http://felix.abecassis.me/2011/10/opencv-rotation-deskewing/

Ahmed, N. (2005). *Bangla Optical Character Recognition*. Recuperado de: http://dspace.bracu.ac.bd:8080/xmlui/bitstream/handle/10361/64/Bangla%20Optical%20Character%20Recognition%202.pdf?sequence=1yisAllowed=y

Ajami, S., y Bagheri-Tadi, T. (2013). *Barriers for Adopting Electronic Health Records (EHRs) by Physicians*. Recuperado de: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3766548/

Arora, S., Malik, L., Bhattacharjee, D., y Nasipuri, M. (2010). *Classification Of Gradient Change Features Using MLP For Handwritten Character Recognition*. Recuperado de: https://arxiv.org/ftp/arxiv/papers/1006/1006.5927.pdf

Auld, A. (2016). *Do you speak the language of data analytics*. Recuperado de: https://www.cbronline.com/big-data/analytics/speak-language-data-analytics/

Ayala, E., Garcia, J. y Guzmán, J. (2007) *Prototipo de sistema informático para la toma de decisiones, manejo de expedientes médicos y control de citas de pacientes para hospital primero de mayor*. Recuperado de: http://biblio.udb.edu.sv/library/index.php?title=96906ylang=enyquery=@title=Special:GSMSearchPage@process=@autor=AYALA%20MORALES,%20EDUARDO%20JAVIER@mode=yrecnum=1ymode=

Azure ML, Team, (2015). *Azure ML Text Classification template*. Recuperado de: https://blogs.technet.microsoft.com/machinelearning/2015/05/06/azure-ml-text-classification-template/

Biondich, P., Overhage, M., Dexter, P., Downs, S., Lemmon, L., y McDonald, C. (2002). *A modern optical character recognition system in a real world clinical setting:*

some accuracy and feasibility observations. Recuperado de: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2244242/

Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python.* O'Reilly Media Inc. Recuperado de: http://victoria.lviv.ua/html/fl5/Natural LanguageProcessingWithPython.pdf

Bishop, C. (Enero, 2015). OCR? ICR? IWR? OMG! Get the Most from Your Scanned Text. Recuperado de: http://www.thecrowleycompany.com/ocr-icr-iwr-omg-get-scanned-text/

Bushinak, H., AbdelGaber, S., y AlSharif, F. (2011). *Recognizing The Electronic Medical Record Data From Unstructured Medical Data Using Visual Text Mining Techniques.* Recuperado de International Journal of Computer Science and Information Security9.6: 25-35.: http://search.proquest.com/openview/fbe73ae997b0b1ec04daf2c0be95631c/1?pq-origsite=gscholarycbl=616671

Cama Castillo, Y. A. (2015). Prototipo computacional para la detección y clasificación de expresiones faciales mediante la extracción de patrones binarios locales. Pontificia Universidad Católica del Peru. Recuperado de: http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/5960/CAMA_YULIAN_PROTOTIPO_COMPUTACIONAL.pdf?sequence=1yisAllowed=y

Candice, N., y Erasmus, L. (2016). *Electronic Medical Records: A developing and developed country analysis*. Recuperado de: http://iamot2016.org/proceedings/papers/IAMOT_2016_paper_32.pdf

Carbonell J. (1992). *El procesamiento del lenguaje natural, tecnología en transición*. Congreso de la Lengua Española, Sevilla. Recuperado: https://cvc.cervantes.es/obref/congresos/sevilla/tecnologias/ponenc_carbonell.htm

Castells, M. (1999). The Information Age, Volumes 1-3: Economy, Society and Culture. Cambridge (Mass.); Oxford: Wiley-Blackwell. Recuperado de: https://deterritorialinvestigations.files.wordpress.com/2015/03/manuel_castells_the_rise_of_the_network_societybookfi-org.pdf

Cellan-Jones R. (Diciembre, 2014). Stephen Hawking warns artificial intelligence could end mankind. Recuperado de: http://www.bbc.com/news/technology-30290540

Charles, D., Meghan, G., y Searcy, T. (Abril, 2015). *Adoption of Electronic Health Record Systems among U.S. NonFederal*. Recuperado de: https://www.healthit.gov/sites/default/files/data-brief/2014HospitalAdoptionDataBrief.pdf

Ciampa, M., y Revels, M. (2013). *Introduction to Healthcare Information Technology*. Recuperado de https://www.cengagebrain.com.au/content/9781285244464.pdf

Dadhania, S., y Dhobi, J. (2012). *Improved kNN Algorithm by Optimizing Cross-validation. International Journal of Engineering Research y Technology (IJERT)* Vol. 1 Issue 3, 1 - 6. Recuperado de: http://www.ijert.org/download/135/improved-knn-algorithm-by-optimizing-cross-validation

De la Escalera, A., y Armingol, J. (2010). *Sistemas de Percepción*. Recuperado de: http://ocw.uc3m.es/ingenieria-de-sistemas-y-automatica/sistemas-de-percepcion/transparencias-clase/1-Introduccion.pdf

De la Escalera, A. (2011). *Visión por Computador. Fundamentos y métodos.* Madrid: Prentice Hall. Recuperado de Visión por Computador. Fundamentos y métodos.

Delen, D. (2016). *Applied Data Mining for Business Analytics*. Recuperado de: https://www.safaribooksonline.com/library/view/applied-data-mining/9780134212524/

Eikvil, L. (1993). *Optical Character Recognition*. Recuperado de: https://www.nr.no/~eikvil/OCR.pdf

El Comercio. (Mayo, 2014). *Lolimsa: Solo el 11% de las historias clínicas son virtuales.* Recuperado de: http://elcomercio.pe/economia/negocios/lolimsa-solo-11-historias-clinicas-son-virtuales-noticia-1732352

El Peruano (2015). *Aprueba el Reglamento de la Ley N° 30024, Ley que crea el Registro Nacional de Historias Clínicas Electrónicas*. Recuperado de: http://busquedas.elperuano.com.pe/normaslegales/aprueba-el-reglamento-de-la-ley-n-30024-ley-que-crea-el-re-decreto-supremo-n-039-2015-sa-1324291-4/

Ghaaliq , A., y McCluskey, A. (2008). *Clinical tests: sensitivity and specificity.* Continuing Education in Anaesthesia Critical Care y Pain, Volume 8, Issue 6, 221-223. Recuperado de: https://academic.oup.com/bjaed/article/8/6/221/406440

Grant, E. (2015). *Introduction to Machine Learning*. Recuperado de: http://www.cs.toronto.edu/~urtasun/courses/CSC411/tutorial3.pdf

Grother, P., y Hanaoka, K. (2016). *NIST Special Database 19 Handprinted Forms and Characters 2nd Edition.* Recuperado de National Institute of Standards and Technology: https://s3.amazonaws.com/nist-srd/SD19/sd19_users_guide_edition_2.pdf

Guo, Z., y Hall, W. (Marzo, 1989). *"Parallel Thinning with Two-Subiteration Algorithm",* Communications of the ACM, vol. 32, no. 3. Recuperado de: http://stackoverflow.com/questions/8080383/a-fast-thinning-algorithm

Hayri, S. (2006). *Evaluation of an electronic medical record system: Zonguldak Karaelmas University Hospital Survey*. Recuperado de: https://etd.lib.metu.edu.tr/upload/12608125/index.pdf

HealthIT. (2016). *Basics of Health IT: Health IT Terms.* Recuperado de: https://www.healthit.gov/patients-families/health-it-terms

Hilbert, M. (2015*). Digital Technology and Social Change [Open Online Course at the University of California] (freely available).* Recuperado de: https://canvas.instructure.com/courses/949415

Hu, J., Brown, M., y Turin, W. (2000). *Handwriting Recognition with hidden markov models and grammatical constraints*. Recuperado de IBM Research: http://www.research.ibm.com/people/j/jyhu/iwfhr94.ps

Hung, K., Luk, R., Yeung, D., Chung, K., y Shu, W. (Setiembre, 2002). *Detection of Language (Model) Errors.* Recuperado de: http://www.aclweb.org/anthology/W00-1311

Jack Copeland (2000). *Reference Articles on Turing. What is Artificial Intelligence?*. Recuperado de: http://www.alanturing.net/turing_archive/pages /reference%20articles/what%20is%20ai.html

Lyons, R., Payne, C., McCabe, M. y Fielder, C. (1998). *Legibility of doctors' handwriting: Quantitative comparative study*. Recuperado de: https://www.researchgate.net/publication/13538099_Legibility_of_doctors'_handwriting_Quantitative_comparative_study

Math4IQB. (Octubre, 2013). *APM-Metrics: Confusion Matrix*. Recuperado de: https://www.youtube.com/watch?v=Rzg-rpWLubM

Mathias, J., Agrawal, A., Feinglass, J., Cooper, A., William, D., y Choudhary, A. (2013). *Development of a 5 year life expectancy index in older adults using predictive mining of electronic health record data*. Recuperado de: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3715340/

Menachemi, N., y Collum, T. (2011). *Benefits and drawbacks of electronic health record systems.* Recuperado de: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3270933/

Merriam-Webster. (2016). *Lexicon*. Recuperado de: http://www.merriam-webster.com/dictionary/lexicon

Mickevicius, V., Krilavicius, T. y Morkevicius, V. (2015). *Classification of short legal lithuanian texts.* Recuperado de: http://bpti.lt/wp-content/uploads/2016/02/bsnlp2015.pdf

Niklas, F. (2016). *Image Processing.* Recuperado de: http://felixniklas.com/imageprocessing/binarization

La República (Diciembre, 2010). *Los avances tecnológicos que marcaron el 2010*. Recuperado de: http://larepublica.pe/21-12-2010/los-avances-tecnologicos-que-marcaron-el-2010

Landini, G. (2016). *Auto Threshold*. Recuperado de: http://imagej.net/Auto_Threshold#IsoData

Lei He, C., Ping, Z., Jianxiong, D., Ching , Y., y Tien, D. (2012). *The Role of Size Normalization on the Recognition Rate of Handwritten Numerals.* Montreal, Quebec, Canada. Recuperado de: https://www.researchgate.net/publication/242102821_The_Role_of_Size_Normalization_on_the_Recognition_Rate_of_Handwritten_Numerals

LeCun, Y., Cortes, C., y Burges, C. (Noviembre, 1998). *The MNIST database of handwritten digits*. Recuperado de: http://yann.lecun.com/exdb/mnist/

Old Dominion University. (2010). *Recall Precision*. Recuperado de: http://www.cs.odu.edu/~mukka/cs795sum10dm/Lecturenotes/Day4/recallprecision.docx

Ouchtati, S., Redjimi, M., y Bedda, M. (2015). *An Offline System for the Recognition of the Fragmented Handwritten Numeric Chains.* International Journal of Future Computer and Communication, Vol. 4, No. 1, 33-39. Recuperado de: http://www.ijfcc.org/vol4/351-C032.pdf

Outomuro, D. (2013). *Estimación del tiempo de consulta*. Recuperado de: www.scielo.cl/pdf/rmc/v141n3/art12.pdf

Ovidiu Ivanciuc (2007), *Applications of Support Vector Machines in Chemistry*. Reviews in Computational Chemistry, Volumen 23. pp 291-400. Recuperado de: http://www.ivanciuc.org/Files/Reprint/Ivanciuc_SVM_CCR_2007_23_291.pdf

Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., Duchesnay E. (2011). *Scikit-learn: Machine Learning in Python*. Recuperado de: http://jmlr.org/papers/v12/pedregosa11a.html.

Perea, J., Martín, M., Montejo, A., Diaz, M. (2008). *Categorización de textos biomédicos usando UMLS*. Procesamiento del Lenguaje Natural, Revista n° 40 121- 127. Recuperado de http://www.sepln.org/revistaSEPLN/revista/40/todo.pdf

Perez A., Casillas A., Gojenola K., Oronoz M., Aguirre N. y Amillano E. (2015). *Detección de fármacos genéricos en textos biomédicos, Procesamiento del Lenguaje Natural,* Revista n° 53 77- 84. Recuperado de: http://www.sepln.org/revistaSEPLN/revista/53/todo.pdf

Pestana Delgado, Roberto et al. (2005). *Concordancia entre el diagnóstico médico y la codificación de informática, considerando el CIE-10, en la consulta externa de pediatría en el Hospital Nacional Cayetano Heredia*, Lima-Perú. Recuperado de: http://www.scielo.org.pe/scielo.php?script=sci_arttextypid=S1018-130X2005000400003ylng=esynrm=iso

Plamondon, R. (2000). *On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey*. Recuperado de Center of excellence for document analysis and recognition, University at Buffalo: http://www.cedar.buffalo.edu/papers/articles/Online_Offline_2000.pdf

Pradeep, J., Srinivasan, E., y Himavathi, S. (Marzo, 2011). *Neural Network based Handwritten Character Recognition system without feature extraction*. International Conference on Computer, Communication and Electrical Technology – ICCCET 2011, 18th y 19th March, (pp. 40-44). Recuperado de: http://ieeexplore.ieee.org/document/5762513/

Poovizhi, P. (2014). *A Study on Preprocessing Techniques for the character recognition*. Recuperado de: https://www.google.com.pe/url?sa=tyrct=jyq=yesrc=sysource=webycd=6ycad=r

jayuact=8yved=0ahUKEwjxgsOj5PrOAhXCW5AKHbAFBf0QFghGMAUyurl=http%3A%2F%2Fcyberleninka.ru%2Farticle%2Fn%2Fa-study-on-preprocessing-techniques-for-the-character-recognition.pdfyusg=AFQjC

Rasmussen, Luke V; Peissig, Peegy; McCarty, Catherine; Starren, Justin. (Junio, 2012). *Development of an optical character recognition pipeline for handwritten form fields from an electronic health record*. Recuperado de: https://jamia.oxfordjournals.org/content/19/e1/e90

Revorredo, J., y Cavalcanti, J. (Mayo, 2014*). Una experiencia de implementación del registro*. Recuperado de: http://www.paho.org/journal/index.php?option=com_docmanyview=downloadycategory_slug=pdfs-may-june-2014yalias=744-una-experiencia-de-implementacion-del-registro-medico-electronico-en-peruyItemid=847

Robertson, J. (Junio, 2013). *Top 10 Countries Where Doctors Go Digital*. Recuperado de: http://www.bloomberg.com/slideshow/2013-06-25/top-10-countries-where-doctors-go-digital.html

Rouse, M. (2015). *Health IT (health information technology).* Recuperado de: http://searchhealthit.techtarget.com/definition/Health-IT-information-technology

Sauvola, J., y PietikaKinen, M. (1998). *Adaptive document image binarization.* Recuperado de: http://www.ee.oulu.fi/mvg/files/pdf/pdf_24.pdf

Sanchez, C., y Sandonis, V. (2009). *Reconocimiento Óptico de Caracteres (OCR).* Recuperado de: http://www.it.uc3m.es/jvillena/irc/practicas/08-09/09.pdf

Sancho, F., (2015). *Introducción al aprendizaje automático.* Recuperado de: http://www.cs.us.es/~fsancho/?e=75

Sandin, G. (2015). *Análisis e implementación de algoritmos en nuevas tecnologías de paralelización.* Recuperado de: http://diposit.ub.edu/dspace/bitstream/2445/67785/3/memoria.pdf

Sanghamitra, M., y Nandini Das Bebartta, H. (2011). *Performance Comparison of SVM and K-NN for Oriya Character Recognition. (IJACSA)* International Journal of Advanced Computer Science and Applications, Special Issue on Image Processing and Analysis, 112-116. Recuperado de: https://thesai.org/Downloads/SpecialIssueNo1/Paper_16-Performance%20Comparison%20of%20SVM%20and%20K-NN%20for%20Oriya%20character%20recognition.pdf

Scikit-learn (2016). *Machine Learning in Python.* Recuperado de: http://scikit-learn.org/stable/index.html

Segura I., Martínez P., Samy D. (2008). *Detección de fármacos genéricos en textos biomédicos*. Procesamiento del Lenguaje Natural, Revista n° 40 27- 34. Recuperado de: http://www.sepln.org/revistaSEPLN/revista/40/todo.pdf

Shekelle, P., Morton, S., y Keeler, E. (Abril, 2006). *Costs and Benefits of Health Information Technology. Rockville (MD): Agency for Healthcare Research and Quality (US)*. Recuperado de: http://www.ncbi.nlm.nih.gov/books/NBK37992/

Simborg, D. W. (2008). *Promoting Electronic Health Record Adoption. Is It the Correct Focus?* Journal of the American Medical Informatics Association : JAMIA, 127-129. Recuperado de: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2274790/

Singh, H., y Sitting, D. (2015). *Measuring and improving patient safety through health information technology: The Health IT Safety Framework*. Recuperado de: http://qualitysafety.bmj.com/content/early/2015/09/13/bmjqs-2015-004486.full.pdf+html

Smith, B. (2011). *Improving US Life Expectancy through EMR Technology*. Recuperado de: http://en.community.dell.com/dell-blogs/direct2dell/b/direct2dell/archive/2011/06/14/improving-us-life-expectancy-through-emr-technology-infographic

Smith, R. (2016). *An Overview of the Tesseract OCR Engine*. Recuperado de: http://static.googleusercontent.com/media/research.google.com/en//pubs/archive/33418.pdf

Smola, A. & Vishwanathan S. (2008). *Introduction to machine learning*. Recuperado de: https://www.kth.se/social/upload/53a14887f276540ebc81aec3/online.pdf

Smythe, R. (2014). *Forbes: Pharma y Healthcare. Why Changing Health Care Is Hard.* Recuperado de: http://www.forbes.com/sites/roysmythe/2014/02/24/why-changing-health-care-is-hard/#10971176407b

Statnikov, A., Hardin, D., Guyon, I. y Aliferis, C. (2009). *A gentle introduction to support vector machines in biomedicine.* Recuperado de: https://med.nyu.edu/chibi/sites/default/files/chibi/Final.pdf

Stone, C. (2014). *A Glimpse at EHR Implementation Around the World: The Lessons the US Can Learn*. Recuperado de: http://www.e-healthpolicy.org/docs/A_Glimpse_at_EHR_Implementation_Around_the_World1_ChrisStone.pdf

Storcheus, D. (2015). *Google Research*. Recuperado de: http://www.slideshare.net/g33ktalk/dataengconf-feature-extraction-modern-questions-and-challenges-at-google

Stromme, A., y Carlson, R. (2010). *Minimally Supervised Methods to Correct Optical Character Recognition.* Recuperado de: Carnegie Mellon University School of Computer Science: http://www.cs.cmu.edu/~rcarlson/docs/RyanCarlson_nlp.pdf

Sucar, L. E., y Gomez, G. (2011). *Vision Computacional*. Recuperado de: http://ccc.inaoep.mx/~esucar/Libros/vision-sucar-gomez.pdf

Sun, H. (2015). *k− Nearest Neighbour and SVM classifier with feature extraction and feature selection*. Recuperado de: http://homepages.rpi.edu/~sunh6/15fall6967.pdf

The University of Utah. (2012). *College of Engineering.* Recuperado de: http://www.coe.utah.edu/~cs4640/slides/Lecture11.pdf

Turing, A.M. (1950). *Computing machinery and intelligence*. Mind, 59, 433-460. Recuperado de: http://loebner.net/Prizef/TuringArticle.html

UC3M. (2016). *UC3Mx: ISA.1x Introducción a la visión por computador: desarrollo de aplicaciones con OpenCV*. Recuperado de: https://courses.edx.org/courses/course-v1:UC3Mx+ISA.1x+1T2016/courseware/1b3mlg101867bi11isc1jeq1duu4/1b3mlg101cvghei1jefrhdmv8b/

University Alliance. (2016). *Benefits of Electronic health Records*. Recuperado de: http://www.usfhealthonline.com/resources/healthcare/benefits-of-ehr/#.V2NrzfkrKVN

Vamvakas, G. (2014). *Optical Character Recognition for Handwritten Characters*. Recuperado de: http://www.slideshare.net/lovebot/off-line-handwritten-optical-character-regonisation-ocr

Vashisht, K., y Nandal, N. (Marzo, 2016). *International Journal of Scientific Engineering and Applied Science (IJSEAS)* – Volume-2, Issue-3. Recuperado de: http://ijseas.com/volume2/v2i3/ijseas20160305.pdf

VeryPDF. (2015). *VeryPDF Image Processing SDK, Automatically clean-up images, including auto-rotation, auto-deskew, crop, noise removal, etc. operations*. Recuperado de: http://www.verypdf.com/wordpress/201511/verypdf-image-processing-sdk-automatically-clean-up-images-including-auto-rotation-auto-deskew-crop-noise-removal-etc-operations-42022.html

Wargemann, P. (2003). *EHR vs. CPR vs. EMR*. Recuperado de: http://www.providersedge.com/ehdocs/ehr_articles/EHR_vs_CPR_vs_EMR.pdf

Wilson, P., y McEvoy, S. (2011). *Health IT JumpStart*. Hoboken, US: Sybex. Recuperado de http://www.ebrary.com

World Health Organization (1993). *The ICD-10 Classification of Mental and Behavioural Disorders, Diagnostic criteria for research*. Recuperado de: http://apps.who.int/iris/bitstream/10665/37108/1/9241544554.pdf.

Yan Tam, K., y Y. Kiang, M. (1992). *Managerial Applications of Neural Networks: The Case of Bank Failure Predictions.* Management Science, 926 - 947. Recuperado de: https://www.jstor.org/stable/2632376

Zhu, X. (2005). *K-nearest-neighbor: an introduction to machine learning*. Recuperado de: http://www.cs.sun.ac.za/~kroon/courses/machine_learning/lecture2 /kNN-intro_to_ML.pdf

BIBLIOGRAFÍA

Agile Modeling. (2018). *UML 2 Component Diagrams: An Agile Introduction*. Recuperado de: http://www.agilemodeling.com/artifacts/componentDiagram.htm

Chang, Chih-Chung, and Chih-Jen Lin (2001), *LIBSVM: a library for support vector machines*. Recuperado de: http://www.csie.ntu.edu.tw/!cjlin/libsvm

Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin (2016). *A Practical Guide to Support Vector Classification.* Recuperado de: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

International Resources Management Association. (2013). *User-Driven Healthcare: Concepts, Methodologies, Tools and Applications. USA: IRMA International.* Recuperado de: https://books.google.com.pe/books?id=mcSeBQAAQBAJyprintsec=frontcoverydq=UserDriven+Healthcare:+Concepts,+Methodologies,+Tools+and+Applicationsyhl=enysa=Xyved=0ahUKEwiq7cqSk8jNAhVSfiYKHYVWCC4Q6AEIHDAA#v=onepageyq=User-Driven%20Healthcare%3A%20Concepts%2C

Joachims, T. (1998). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features.* Recuperado de: https://link.springer.com/chapter/10.1007/BFb0026683

Ministerio de Salud del Perú. (2013). *Registro Nacional de Historias Clínicas Electrónicas*. Recuperado de: http://www.minsa.gob.pe/renhice/?op=1

Ministerio de sanidad, servicios sociales e igualdad de España. (2017) *Manual de Codificación CIE-10-ES Diagnósticos.* Recuperado de: http://www.msssi.gob.es/estadEstudios/estadisticas/normalizacion/CIE10/Diagnost_CIE10ES_2017.pdf

MIT OpenCourseWare (2014). *Artificial Intelligence: 16. Learning: Support Vector Machines [Video File]*. Recuperado de: https://www.youtube.com/watch?v=_PwhiWxHK8o

Nordlandle, E. (2001). *AI surveying: Artificial intelligence in business*. Recuperado de: http://www.4c.ucc.ie/web/upload/publications/mastersThesis/Artificial_Intelligence_in_Business.pdf

OpenCV. (2016). *OpenCV 2.4.13.2 documentation*. Recuperado de: https://docs.opencv.org/2.4.13.2/

Rennie, J.D.M. and R. Rifkin. (2001). *Improving Multiclass Text Classification with the Support Vector Machine*. Recuperado de: http://people.csail.mit.edu/jrennie/papers/aimemo2001.pdf

Shafar, B. (2015). *How to Digitize Texts with Open-Source Command-Line Optical Character Recognition (OCR) Software*. Recuperado de: https://hdw.artsci.wustl.edu/articles/154

The Stanford Natural Language Processing Group (2015). *Spanish FAQ for Stanford CoreNLP, parser, POS tagger, and NER.* Recuperado de: https://nlp.stanford.edu/software/spanish-faq.shtml

Zhu, Y., Hernandez, L., Mueller, P., Dong, Y., y Forman, M. (2013). *Data Acquisition and Preprocessing in Studies on Humans: What Is Not Taught in Statistics Classes?* Am Stat., 67(4): 235–241. Recuperado de: https://www.ncbi.nlm.nih.gov/pubmed/24511148

.

ANEXOS

Anexo 1: Cross-validation por número de vecinos cercanos sin normalizar

K- fold cross-validation sin normalización, a la derecha son los tiempos en procesar los 200 caracteres y en la parte inferior con el tiempo en cargar las clasificaciones a memoria.

K=1					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.79	200	158	42	40.069
3	0.8	200	160	40	34.959
5	0.79	200	158	42	35.417
7	0.795	200	159	41	35.085
9	0.795	200	159	41	36.046
11	0.785	200	157	43	34.66
13	0.775	200	155	45	36.002
15	0.78	200	156	44	38.691
17	0.78	200	156	44	42.216
19	0.775	200	155	45	42.793
21	0.76	200	152	48	42.535

Tiempo(s) 40.108

K=2					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.765	200	153	47	41.455
3	0.78	200	156	44	35.549
5	0.79	200	158	42	34.782
7	0.77	200	154	46	34.559
9	0.785	200	157	43	35.21
11	0.775	200	155	45	34.829
13	0.76	200	152	48	34.58
15	0.75	200	150	50	34.734
17	0.755	200	151	49	34.915
19	0.745	200	149	51	34.877
21	0.74	200	148	52	34.808

Tiempo(s) 35.321

K=3					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.695	200	139	61	41.09
3	0.725	200	145	55	34.978
5	0.715	200	143	57	35.412
7	0.73	200	146	54	34.796
9	0.72	200	144	56	34.671
11	0.715	200	143	57	34.955
13	0.75	200	150	50	35.455
15	0.75	200	150	50	35.267
17	0.73	200	146	54	34.93
19	0.73	200	146	54	34.987
21	0.735	200	147	53	35.101

Tiempo(s) 34.121

K=4					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.75	200	150	50	38.37
3	0.76	200	152	48	35.436
5	0.76	200	152	48	35.436
7	0.745	200	149	51	35.066
9	0.75	200	150	50	35.072
11	0.74	200	148	52	34.695
13	0.755	200	151	49	35.047
15	0.75	200	150	50	34.72
17	0.745	200	149	51	34.804
19	0.745	200	149	51	35.207
21	0.75	200	150	50	35.496

Tiempo(s) 36.888

K=5					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.73	200	146	54	39.685
3	0.76	200	152	48	35.264
5	0.77	200	154	46	34.893
7	0.75	200	150	50	35.017
9	0.74	200	148	52	34.841
11	0.745	200	149	51	35.023
13	0.74	200	148	52	34.916
15	0.745	200	149	51	35.278
17	0.755	200	151	49	34.926
19	0.755	200	151	49	35.064
21	0.735	200	147	53	35.193

Tiempo(s) 34

K=6					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.765	200	153	47	41.085
3	0.78	200	156	44	35.398
5	0.84	200	168	32	35.021
7	0.795	200	159	41	34.792
9	0.79	200	158	42	34.674
11	0.785	200	157	43	34.701
13	0.78	200	156	44	35.147
15	0.775	200	155	45	40.709
17	0.76	200	152	48	36.357
19	0.76	200	152	48	36.317
21	0.745	200	149	51	35.381

Tiempo(s) 33.403

K=7					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.755	200	151	49	44.944
3	0.77	200	154	46	37.77
5	0.75	200	150	50	36.344
7	0.765	200	153	47	36.14
9	0.745	200	149	51	36.477
11	0.75	200	150	50	35.834
13	0.73	200	146	54	34.994
15	0.74	200	148	52	35.334
17	0.745	200	149	51	34.618
19	0.73	200	146	54	34.889
21	0.73	200	146	54	34.737

Tiempo(s) 36.248

K=8					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.77	200	154	46	40.501
3	0.725	200	145	55	34.616
5	0.74	200	148	52	34.883
7	0.755	200	151	49	34.812
9	0.765	200	153	47	35.13
11	0.74	200	148	52	34.86
13	0.75	200	150	50	35.323
15	0.755	200	151	49	40.08
17	0.75	200	150	50	40.308
19	0.725	200	145	55	40.66
21	0.715	200	143	57	38.083

Tiempo(s) 36.248

K=9					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.72	200	144	56	45.719
3	0.72	200	144	56	35.041
5	0.72	200	144	56	35.039
7	0.73	200	146	54	34.862
9	0.74	200	148	52	35.03
11	0.72	200	144	56	34.801
13	0.725	200	145	55	38.922
15	0.71	200	142	58	51.388
17	0.7	200	140	60	53.39
19	0.715	200	143	57	52.815
21	0.715	200	143	57	54.476

Tiempo(s) 33.875

K=10					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.785	200	157	43	36.088
3	0.785	200	157	43	35
5	0.795	200	159	41	34.784
7	0.78	200	156	44	34.691
9	0.785	200	157	43	34.886
11	0.8	200	160	40	34.759
13	0.785	200	157	43	34.792
15	0.785	200	157	43	34.921
17	0.785	200	157	43	34.601
19	0.78	200	156	44	34.859
21	0.78	200	156	44	34.837

Tiempo(s) 34.344

K=11					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.825	200	165	35	41.286
3	0.815	200	163	37	35.505
5	0.855	200	171	29	34.661
7	0.835	200	167	33	35.032
9	0.83	200	166	34	35.036
11	0.83	200	166	34	35.543
13	0.835	200	167	33	35.243
15	0.835	200	167	33	34.912
17	0.845	200	169	31	35.122
19	0.83	200	166	34	34.98
21	0.82	200	164	36	34.818

Tiempo(s) 34.422

Anexo 2: Cross-validation por número de vecinos cercanos con normalización

K-fold cross-validation con normalización, a la derecha son los tiempos en procesar los 200 caracteres y en la parte inferior con el tiempo en cargar las clasificaciones a memoria.

	K=1				
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.83	200	166	34	49.038
3	0.84	200	168	32	44.466
5	0.865	200	173	27	45.163
7	0.855	200	171	29	47.184
9	0.835	200	167	33	46.484
11	0.835	200	167	33	45.208
13	0.83	200	166	34	43.927
15	0.83	200	166	34	43.872
17	0.835	200	167	33	44.821
19	0.835	200	167	33	44.188
21	0.82	200	164	36	45.395

Tiempo(s) 31.956

K=2					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.82	200	164	36	50.917
3	0.81	200	162	38	42.588
5	0.83	200	166	34	41.986
7	0.82	200	164	36	34.846
9	0.85	200	170	30	41.866
11	0.82	200	164	36	48.53
13	0.83	200	166	34	53.867
15	0.82	200	164	36	41.847
17	0.815	200	163	37	40.759
19	0.825	200	165	35	40.411
21	0.81	200	162	38	41.452

Tiempo(s) 37.972

K=3					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.78	200	156	44	41.888
3	0.795	200	159	41	40.902
5	0.815	200	163	37	40.945
7	0.815	200	163	37	42.862
9	0.835	200	167	33	42.552
11	0.83	200	166	34	41.813
13	0.82	200	164	36	40.007
15	0.805	200	161	39	40.691
17	0.8	200	160	40	41.243
19	0.815	200	163	37	40.217
21	0.815	200	163	37	42.576

Tiempo(s) 35.387

K=4					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.8	200	160	40	45.323
3	0.81	200	162	38	44.673
5	0.845	200	169	31	45.523
7	0.83	200	166	34	44.165
9	0.84	200	168	32	44.303
11	0.83	200	166	34	38.841
13	0.825	200	165	35	38.692
15	0.82	200	164	36	38.576
17	0.82	200	164	36	37.639
19	0.81	200	162	38	37.435
21	0.83	200	166	34	37.539

Tiempo(s) 39.945

K=5					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.825	200	165	35	45.942
3	0.83	200	166	34	38.532
5	0.83	200	166	34	37.446
7	0.805	200	161	39	37.508
9	0.805	200	161	39	37.493
11	0.82	200	164	36	37.144
13	0.815	200	163	37	36.65
15	0.805	200	161	39	36.508
17	0.805	200	161	39	34.975
19	0.81	200	162	38	34.931
21	0.81	200	162	38	35.199

Tiempo(s) 39.176

K=6					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.79	200	158	42	41.495
3	0.825	200	165	35	35.635
5	0.84	200	168	32	35.035
7	0.83	200	166	34	34.514
9	0.82	200	164	36	34.953
11	0.815	200	163	37	37.239
13	0.805	200	161	39	37.884
15	0.795	200	159	41	36.743
17	0.8	200	160	40	37.914
19	0.8	200	160	40	38.508
21	0.795	200	159	41	39.347

Tiempo(s) 33.403

K=7					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.81	200	162	38	46.305
3	0.805	200	161	39	40.516
5	0.79	200	158	42	39.059
7	0.8	200	160	40	39.505
9	0.805	200	161	39	36.301
11	0.79	200	158	42	35.114
13	0.8	200	160	40	34.725
15	0.79	200	158	42	34.879
17	0.78	200	156	44	37.761
19	0.775	200	155	45	34.927
21	0.775	200	155	45	34.761

Tiempo(s) 37.741

K=8					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.85	200	170	30	51.547
3	0.82	200	164	36	40.546
5	0.835	200	167	33	37.833
7	0.83	200	166	34	37.606
9	0.82	200	164	36	38.001
11	0.805	200	161	39	37.893
13	0.805	200	161	39	37.13
15	0.795	200	159	41	37.389
17	0.79	200	158	42	37.64
19	0.785	200	157	43	37.47
21	0.775	200	155	45	37.706

Tiempo(s) 37.75

K=9					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.81	200	162	38	48.323
3	0.85	200	170	30	37.374
5	0.85	200	170	30	37.159
7	0.875	200	175	25	37.171
9	0.855	200	171	29	37.254
11	0.85	200	170	30	37.446
13	0.83	200	166	34	37.101
15	0.83	200	166	34	38.008
17	0.82	200	164	36	37.707
19	0.825	200	165	35	37.177
21	0.82	200	164	36	37.68

Tiempo(s) 38.406

K=10					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.85	200	170	30	41.899
3	0.855	200	171	29	38.039
5	0.855	200	171	29	37.378
7	0.835	200	167	33	37.366
9	0.845	200	169	31	39.05
11	0.83	200	166	34	38.464
13	0.82	200	164	36	37.66
15	0.815	200	163	37	37.729
17	0.805	200	161	39	37.813
19	0.815	200	163	37	37.75
21	0.8	200	160	40	37.489

Tiempo(s) 38.47

K=11					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.845	200	169	31	49.518
3	0.88	200	176	24	39.985
5	0.885	200	177	23	39.149
7	0.9	200	180	20	38.773
9	0.895	200	179	21	37.829
11	0.89	200	178	22	37.591
13	0.885	200	177	23	37.974
15	0.895	200	179	21	37.594
17	0.88	200	176	24	37.866
19	0.875	200	175	25	38.369
21	0.87	200	174	26	37.677

Tiempo(s) 39.243

Anexo 3: Matriz de confusión

Matriz confusión original, incluye cada letra con su denominación en ASCII.

Actual/Obtenido	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122
65	171	0	0	1	0	1	0	3	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0	0	0	0	1	0	0	0	0	3	0	0	0	0	0	0	0	12	0	2	0	2	0	0	0	0	0	1	0	0
66	2	166	3	5	0	0	1	0	4	0	0	0	0	0	2	6	0	1	0	1	0	0	0	0	0	0	1	2	0	0	2	0	0	0	1	0	0	1	0	0	0	0	2	0	0	1	0	0	0	0	0	0
67	0	0	132	0	0	0	0	0	1	0	0	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	61	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
68	0	0	0	188	0	0	0	0	1	0	0	0	0	0	7	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0
69	0	0	0	0	183	1	0	0	4	0	0	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	0	0	0	1	0	0	1	0	0	0	0	2	0	1	0	0	0	0	0	0	0
70	0	0	0	0	0	140	0	0	1	0	0	0	0	0	0	4	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	43	0	0	0	0	0	2	0	0	0	5	0	2	0	2	0	0	0	0	0	0
71	0	1	5	0	0	0	183	0	0	0	0	0	0	0	0	1	2	1	0	0	0	0	0	0	0	0	0	1	3	0	0	0	0	0	1	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0
72	4	0	0	0	0	0	0	188	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
73	0	0	0	0	0	0	0	0	159	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	13	1	0	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0
74	0	0	0	1	0	0	0	0	3	162	0	0	0	0	0	0	0	0	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0
75	0	0	0	0	0	1	0	1	2	0	156	2	0	0	0	0	0	0	0	0	1	2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	34	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
76	0	0	1	0	0	0	0	0	2	0	0	195	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
77	0	0	1	0	0	0	0	1	0	0	0	0	138	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	52	4	0	0	0	1	0	0	0	1	0	0	0	0
78	0	0	0	0	0	0	0	4	0	1	0	0	0	193	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
79	0	0	0	1	0	0	0	0	4	0	0	0	0	0	147	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	48	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	1	0	5	0	0	1	0	0	0	0	0	0	132	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	57	0	2	0	0	0	0	0	0	0	0
81	0	0	5	0	0	0	2	0	0	0	0	0	0	0	12	1	156	1	0	0	0	0	0	0	0	0	10	0	0	0	1	0	1	0	0	0	0	0	0	0	8	1	2	0	0	0	1	0	0	0	0	0
82	6	1	0	0	0	0	0	0	1	0	0	1	0	0	0	4	0	181	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
83	0	1	0	0	0	0	0	0	6	0	0	0	0	0	2	0	0	0	137	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	53	0	0	0	0	0	0	0
84	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	196	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2	0	0	0	0	0	0	0
85	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	150	7	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	37	2	0	0	0	0	0
86	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	5	138	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	3	0	0	2	48	0	0	0	0	0
87	0	0	0	0	0	0	0	1	0	1	0	0	1	19	0	0	0	0	0	0	2	4	136	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	33	0	0
88	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	135	9	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	46	5	1
89	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	5	0	0	150	0	0	0	0	0	0	0	0	0	0	1	0	6	0	0	0	0	0	2	0	0	1	0	0	0	33	0
90	0	1	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	1	149	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	44
97	0	1	0	0	0	0	0	0	1	0	0	0	0	0	3	0	2	0	0	0	0	0	0	0	0	0	181	0	0	3	1	0	0	0	1	0	0	0	0	0	2	0	2	0	0	0	1	0	0	0	0	3
98	0	0	1	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	193	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
99	0	0	33	0	0	0	0	0	0	0	0	3	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	159	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	1	3	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	192	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
101	1	0	17	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	179	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
102	0	0	1	0	0	48	0	0	0	1	0	0	0	0	0	6	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	124	0	0	1	0	0	2	0	0	0	3	0	2	0	10	0	0	0	0	0	0
103	1	1	0	0	1	2	0	0	1	15	0	0	0	0	2	1	0	0	2	0	0	0	0	0	0	0	1	0	0	2	0	0	132	0	0	5	0	5	0	0	1	0	21	0	1	1	0	0	0	0	0	1
104	1	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	191	0	0	0	3	0	2	0	0	0	0	1	0	0	0	0	0	0	0
105	0	0	0	0	0	0	0	0	40	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	132	1	0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0
106	0	0	0	0	0	0	0	0	1	14	0	0	0	0	1	0	0	0	0	3	1	0	1	0	0	0	0	0	0	3	0	0	0	0	21	153	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0
107	2	0	0	0	0	1	0	1	1	0	51	11	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4	0	0	124	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0
108	0	0	0	0	0	0	0	0	57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13	1	0	130	0	0	0	0	0	0	0	0	0	0	0	0	0	0
109	1	0	0	0	0	0	0	1	0	0	0	0	49	2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3	0	0	0	0	140	3	0	0	0	1	0	0	0	0	0	0	0	0
110	3	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	194	0	0	0	2	0	0	0	0	0	0	0	0
111	0	0	0	3	0	0	0	0	3	0	0	0	0	0	61	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	2	130	0	0	0	0	0	0	0	0	0	0	0
112	0	0	0	1	0	4	0	0	1	0	0	0	0	1	0	35	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	153	0	2	0	1	0	0	0	0	0	0
113	1	1	1	1	1	0	0	0	2	0	0	1	0	0	0	1	0	0	0	3	0	0	0	8	0	3	2	0	0	2	0	2	0	12	1	1	0	0	2	0	0	0	157	1	0	0	0	0	0	0	0	0
114	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	189	0	0	0	1	0	0	1	0
115	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	144	0	0	0	0	0	0	0
116	0	0	3	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	0	0	0	0	0	3	0	178	0	0	0	0	0	0
117	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	45	4	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	0	0	0	0	141	2	1	0	2	0
118	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	3	46	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	4	140	0	0	0	0
119	0	0	0	0	0	0	0	3	0	0	0	0	0	10	0	0	0	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	168	0	0	0
120	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	34	5	1	0	0	0	0	1	0	0	1	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	148	6	0
121	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	6	0	0	35	0	0	0	0	0	0	0	1	0	1	2	0	1	0	0	0	0	0	0	0	4	1	4	0	0	143	0
122	1	0	0	0	1	0	0	0	0	0	0	3	0	0	0	0	0	0	0	4	0	0	0	1	0	36	0	1	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	149

Anexo 4: Matriz de confusión de letras normalizadas con enderezamiento

Nueva Matriz confusión, incluye cada letra con su denominación en ASCII, hecha a partir de letras normalizadas con enderezamiento

Actual/Obtenido	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122
65	180	0	0	0	0	0	0	3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	12	0	1	0	1	0	2	0	0	0	0	0	0
66	0	174	1	6	0	0	0	0	4	0	0	0	0	0	3	1	0	0	0	1	0	0	0	0	0	1	0	0	1	0	3	0	1	0	0	0	0	1	0	1	1	0	2	0	0	0	0	0	0	0	0	0
67	0	0	143	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	51	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
68	0	0	0	185	0	0	0	0	2	0	0	0	0	0	8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0
69	0	0	0	0	185	0	0	0	6	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	0	0	1	0	0	2	0	0	0	0	0	2	0	0	0	0	0	0	0	0
70	0	0	0	0	0	133	0	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	54	0	0	0	0	0	0	0	0	0	4	0	6	0	1	0	0	0	0	0	0
71	0	1	5	0	0	0	183	0	0	0	0	0	0	0	1	0	2	0	0	0	1	0	0	1	0	0	0	0	2	0	0	1	0	1	0	0	0	0	0	2	0	0	1	0	0	0	0	0	0	0	0	0
72	4	0	0	0	0	0	0	185	0	0	1	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	2	0	0	0	1	0	0	1	0	0	1	1	0
73	0	0	0	0	0	0	0	0	165	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	26	0	0	0	0	0	0	0	0	0	0	0	0	0	1
74	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	3	9	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	31	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0
75	0	0	0	0	0	0	0	1	0	0	163	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	30	0	0	0	0	0	0	0	0	1	0	0	2	0	0	0
76	0	0	3	0	0	0	0	0	0	0	0	195	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
77	0	0	0	0	0	0	0	1	0	0	0	0	140	3	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47	6	0	0	0	3	0	0	0	0	0	0	0	0
78	0	0	0	0	0	0	0	2	0	0	0	0	1	193	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0
79	0	0	1	1	0	0	0	0	4	0	0	0	0	0	142	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	53	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	1	0	2	0	0	1	0	0	0	0	0	0	128	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	65	0	2	0	1	0	0	0	0	0	0	0
81	0	0	0	2	0	0	1	0	0	0	0	0	0	0	10	0	166	0	0	0	0	0	0	0	0	0	11	0	2	0	0	0	3	0	0	0	0	0	0	4	1	1	0	0	0	0	0	0	0	0	0	0
82	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	182	0	0	0	0	0	0	0	0	1	0	0	0	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
83	0	1	0	0	0	0	0	0	8	2	0	0	0	0	0	0	0	0	128	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	0	1	0	0	1	0	1	56	0	0	0	0	0	0	0	0
84	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	197	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0	0
85	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	144	6	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	48	0	0	0	0	0
86	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	1	43	0	0	0	0	0
87	0	0	0	0	0	0	0	0	0	1	0	0	1	11	0	0	0	0	0	0	2	3	125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	53	0	1	0	0	0
88	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	143	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	49	3	1	0
89	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	0	1	0	0	155	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	0	0	1	0	0	0	38	0	0
90	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	45
97	0	0	0	0	0	0	0	0	1	0	0	0	0	0	2	0	3	0	0	0	1	0	1	0	0	1	188	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
98	0	0	2	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	193	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
99	0	0	42	0	0	0	0	0	0	0	0	1	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	1	152	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4	0	0	192	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
101	0	0	16	0	0	0	0	0	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	177	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
102	0	0	3	0	1	44	0	0	0	0	0	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	134	0	0	1	0	0	0	0	0	2	0	2	0	9	0	0	0	1	0	0	0
103	1	0	0	0	0	1	0	0	3	4	0	1	0	0	1	2	0	0	4	2	0	0	0	0	4	0	0	0	2	0	0	1	141	0	0	1	0	1	0	0	1	0	27	0	1	2	0	0	0	1	0	0
104	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	196	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
105	0	0	0	0	0	0	0	46	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	126	3	0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0
106	0	0	0	0	0	0	0	0	2	18	0	0	0	0	2	0	0	0	1	0	0	1	0	0	0	0	0	0	0	2	0	0	0	1	21	150	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0
107	1	0	0	0	0	0	0	2	1	0	54	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3	2	0	131	0	0	0	0	0	0	1	0	0	1	0	0	2	0	0
108	0	0	0	0	0	0	0	0	68	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	122	0	0	0	0	0	0	0	0	0	0	0	0	0	0
109	1	0	0	0	0	0	0	1	0	0	0	0	61	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	129	2	0	0	0	0	0	0	0	1	0	0	0	0
110	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	196	0	2	0	0	0	0	0	0	0	0	0	0
111	0	0	0	2	0	0	0	0	1	0	0	0	0	0	58	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	135	0	0	0	0	0	0	0	0	0	0	0
112	0	0	0	1	0	4	0	0	0	0	0	0	0	1	0	31	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	158	0	1	0	2	0	0	0	0	0	0
113	0	0	1	1	0	0	0	0	1	1	0	1	0	0	0	2	0	0	0	3	0	0	0	0	1	1	2	0	0	0	2	0	7	1	0	0	0	0	0	0	0	0	175	0	0	0	1	0	0	0	0	1
114	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	191	0	0	0	0	0	0	0	0
115	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	63	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	132	0	0	0	0	0	0	0
116	0	0	3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	9	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	187	0	0	0	0	0	0
117	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	49	5	1	0	1	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	137	1	0	0	1	0
118	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	49	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	1	0	0	1	144	0	0	0	0
119	0	0	0	0	0	0	0	3	0	0	0	0	0	3	0	0	0	0	0	0	2	1	17	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	172	0	0	0
120	0	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	1	0	34	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	153	3	0
121	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	2	1	7	0	1	149	0
122	0	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	26	0	0	0	2	1	0	1	1	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	162

Anexo 5: Matriz de confusión con enderezamiento sin considerar el error minúscula/mayúscula

Matriz hecha a partir de letras normalizadas con enderezamiento y sin considerar como error el reconocimiento de letras minúsculas como mayúsculas o viceversa.

Actual/Obtenido	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122
65	180	0	0	0	0	0	0	3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	12	0	1	0	1	0	2	0	0	0	0	0	0
66	0	174	1	6	0	0	0	0	4	0	0	0	0	0	3	1	0	0	0	1	0	0	0	0	0	1	0	0	1	0	3	0	1	0	0	0	0	1	0	1	1	0	2	0	0	0	0	0	0	0	0	0
67	0	0	194	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
68	0	0	0	185	0	0	0	0	2	0	0	0	0	8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0
69	0	0	0	0	188	0	0	0	6	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	2	0	0	0	0	0	2	0	0	0	0	0	0	0	0
70	0	0	0	0	0	187	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	6	0	1	0	0	0	0	0	0
71	0	1	5	0	0	0	183	0	0	0	0	0	0	1	0	2	0	0	0	1	0	0	1	0	0	0	0	0	2	0	0	1	0	1	0	0	0	0	0	2	0	0	1	0	0	0	0	0	0	0	0	0
72	4	0	0	0	0	0	0	187	0	0	1	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	1	0	0	1	1	0
73	0	0	0	0	0	0	0	0	170	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26	0	0	0	0	0	0	0	0	0	0	0	0	0	1
74	0	0	0	0	0	0	0	0	0	184	0	0	0	0	0	0	0	0	3	9	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0
75	0	0	0	0	0	0	0	1	0	0	193	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	0	0	0
76	0	0	3	0	0	0	0	0	0	0	0	196	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
77	0	0	0	0	0	0	0	1	0	0	0	0	187	3	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	3	0	0	0	0	0	0	0	0
78	0	0	0	0	0	0	0	2	0	0	0	0	1	194	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0
79	0	0	1	1	0	0	0	0	4	0	0	0	0	0	195	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	1	0	2	0	0	1	0	0	0	0	0	0	193	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0
81	0	0	0	2	0	0	1	0	0	0	0	0	0	0	10	0	167	0	0	0	0	0	0	0	0	0	11	0	2	0	0	0	3	0	0	0	0	0	0	0	0	4	1	0	0	0	0	0	0	0	0	0
82	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	182	0	0	0	0	0	0	0	0	1	0	0	0	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
83	0	1	0	0	0	0	0	0	8	2	0	0	0	0	0	0	0	0	184	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0
84	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	200	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
85	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	192	6	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
86	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	193	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	1	0	0	0	0	0	0	0
87	0	0	0	0	0	0	0	0	0	1	0	0	1	11	0	0	0	0	0	0	2	3	178	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	0	0	1	0	0	0
88	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	192	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	3	1
89	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	0	1	0	0	193	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	198	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
97	0	0	0	0	0	0	0	1	0	0	0	0	0	2	0	3	0	0	0	0	1	0	0	0	1	0	188	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
98	0	0	2	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	193	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
99	0	0	0	0	0	0	0	0	0	0	0	1	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	194	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4	0	0	192	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
101	0	0	16	0	0	0	0	0	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	177	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
102	0	0	3	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	178	0	0	1	0	0	0	0	0	0	2	0	2	0	9	0	0	0	1	0	0
103	1	0	0	0	0	1	0	0	3	4	0	1	0	0	1	2	0	0	4	2	0	0	0	0	4	0	0	0	0	2	0	1	141	0	0	1	0	1	0	0	1	0	27	0	1	2	0	0	0	0	1	0
104	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	196	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
105	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	172	3	0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0
106	0	0	0	0	0	0	0	0	2	0	0	0	0	2	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	2	0	0	0	1	21	168	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0
107	1	0	0	0	0	0	0	2	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3	2	0	185	0	0	0	0	0	0	1	0	0	1	0	0	2	0	0
108	0	0	0	0	0	0	0	0	68	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	122	0	0	0	0	0	0	0	0	0	0	0	0	0	0
109	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	190	2	0	0	0	0	0	0	0	1	0	0	0	0
110	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	196	0	2	0	0	0	0	0	0	0	0	0	0
111	0	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	193	0	0	0	0	0	0	0	0	0	0	0
112	0	0	0	1	0	4	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	189	0	1	0	2	0	0	0	0	0	0
113	0	0	1	1	0	0	0	0	1	1	0	1	0	0	2	0	0	0	3	0	0	0	0	1	1	2	0	0	0	2	0	7	1	0	0	0	0	0	0	0	0	0	175	0	0	0	1	0	0	0	0	1
114	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	192	0	0	0	0	0	0	0	0
115	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	195	0	0	0	0	0	0
116	0	0	3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	196	0	0	0	0	0
117	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	5	1	0	1	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	196	1	0	0	1
118	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	1	0	0	1	193	0	0	0	0
119	0	0	0	0	0	0	0	3	0	0	0	0	0	3	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	189	0	0
120	0	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	187	3
121	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	2	1	7	0	1	181
122	0	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	0	1	1	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	188

Anexo 6. Holdout cross-validation con adelgazamiento

Holdout cross-validation con adelgazamiento, a la derecha son los tiempos en procesar los 200 caracteres y en la parte inferior con el tiempo en cargar a memoria

K=1					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.36	200	72	128	52.594
3	0.31	200	62	138	36.75
5	0.35	200	70	130	36.446
7	0.33	200	66	134	36.707
9	0.34	200	68	132	35.899
11	0.36	200	72	128	37.592
13	0.36	200	72	128	37.23
15	0.34	200	68	132	36.597
17	0.31	200	62	138	35.736
19	0.32	200	64	136	35.515
21	0.305	200	61	139	35.882
Tiempo(s)	30.743				

.

Anexo 7: Holdout cross-validation con adelgazamiento y enderezamiento

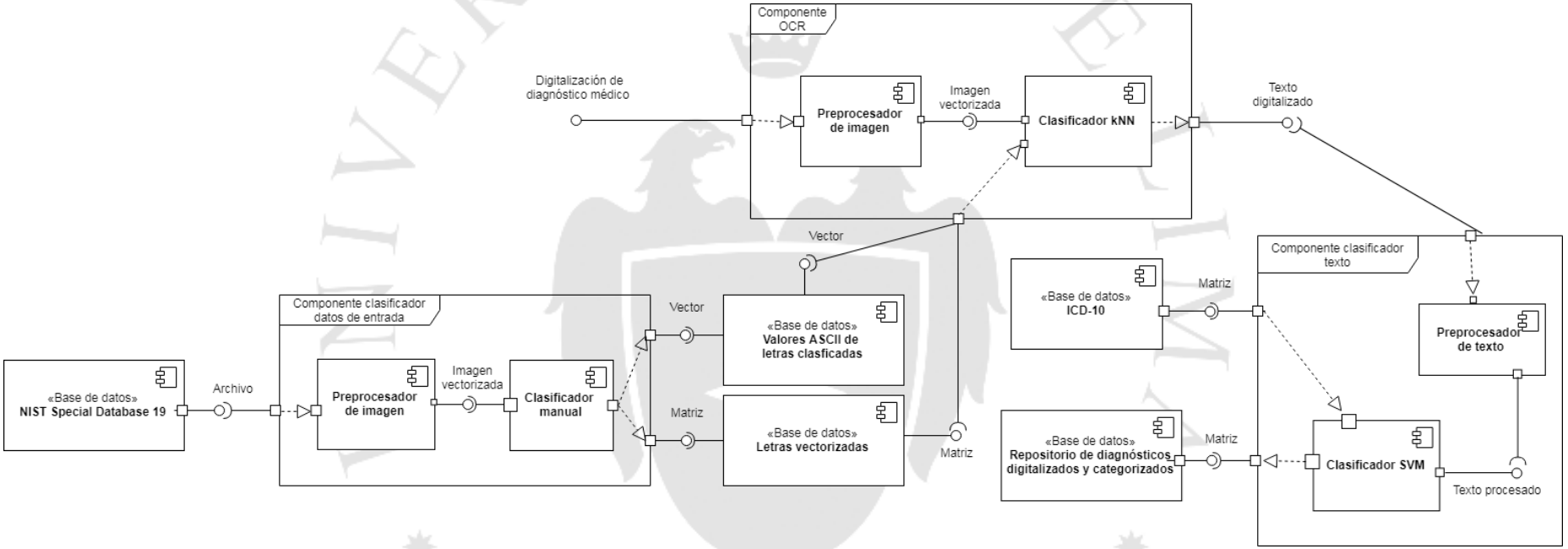
Holdout cross-validation con adelgazamiento y enderezamiento, a la derecha son los tiempos en procesar 200 caracteres y en la parte inferior con el tiempo en cargar.

K=1					
K (vecinos cercanos)	Sensibilidad	Total prueba	Correcto	Incorrecto	Tiempo(s)
1	0.475	200	95	105	48.266
3	0.45	200	90	110	40.522
5	0.51	200	102	98	46.687
7	0.5	200	100	100	39.476
9	0.46	200	92	108	39.285
11	0.47	200	94	106	39.092
13	0.475	200	95	105	39.703
15	0.465	200	93	107	39.162
17	0.47	200	94	106	39.171
19	0.465	200	93	107	37.571
21	0.475	200	95	105	48.266

Tiempo(s) 34.091

Anexo 8: Diagrama de componentes

A continuación, el diagrama de componentes donde se muestra la relación entre distintos elementos de la solución elaborada.



Anexo 9: Librerías utilizadas

1. **OpenCV:** versión 2.4, es una librería con una variedad de funciones enfocadas para el procesamiento de imágenes. Fue utilizada para realizar tanto el preprocesamiento como procesamiento puesto que también permite el uso de algoritmos de machine learning.
2. **Sklearn:** versión 0.19.1, es una librería que contiene varias funciones referentes a machine Learning y minería de datos. A partir de esta librería se extrae la función de SVC, que aplica la técnica de máquinas de soporte vectorial para la clasificación de los registros clínicos.
3. **NLTK:** versión 3.2.5, es una librería que cuenta con distintas funciones para el procesamiento del lenguaje natural. Su utilidad se dio en todo el procesamiento del texto de los diagnósticos antes de iniciar con la categorización de estos.
4. **Mysqldb:** versión 1.2.4, es una librería que permite la conexión a las bases de datos Mysql desde Python. Es parte de la propuesta de solución pues habilita a la solución a hacer consultas e ingresos en la base de datos, siendo el resultado final del proceso de digitalización y categorización.