Optimización del flujo de carga masiva de datos en tecnologías de la información competentes a la telemedicina mediante técnicas OCR (Optical Character Recognition) y procesamiento multinúcleo.

Santiago Ortiz Ceballos

Trabajo de grado presentado para optar al título de Bioingeniero

Asesor
Jonathan Gallego Londoño, Magíster (MSc)

Universidad de Antioquia
Facultad de Ingeniería
Bioingeniería
Medellín, Antioquia, Colombia
2022

Resumen
Las empresas del área de la salud basadas en software que incurren en el área de tecnologías de la información suelen tener una gran cantidad transacciones de datos que superan la capacidad de la memoria RAM de sus equipos, lo cual es un problema comúnmente denominado “Big Data”. Este proyecto intenta mitigar esta problemática aplicando técnicas de reconocimiento de caracteres (OCR) y procesamiento multinúcleo en archivos de tipo imagen y pdf para extraer información precisa sin necesidad de analizar un archivo por completo. Como resultado se obtuvo un proceso eficaz pero lento, tardando 5 segundos en procesar un archivo y cuyo comportamiento es exponencial a medida que el volumen de archivos incrementa. Sin embargo, al extraer información precisa y evitar el bloqueo del hilo principal de la plataforma de software mediante técnicas de multiprocesamiento, se logró optimizar el uso de recursos y la fluidez del servicio, mejorando el rendimiento de la plataforma y la experiencia del usuario que use este tipo de servicios.

Introducción
La pandemia del covid-19 ha forzado la digitalización de muchos sectores en tiempo récord, cambiando las reglas del juego de forma disruptiva e impulsándonos a adoptar actividades remotas en cuanto a educación, trabajo, comunicación, entretenimiento, deporte e incluso la salud. Algunas de estas ya se comenzaban a imaginar virtualmente y se vieron forzadas a ser ejecutadas de súbito como es el caso de la telemedicina.

Afortunadamente, los dispositivos de telecomunicaciones y las tendencias tecnológicas como el Internet de las cosas (IoT), el Big Data, la Inteligencia Artificial y otras, han dado paso a la implementación de sistemas embebidos y herramientas de software que permiten monitorear el estado de un paciente de forma continua y remota, lo que es conocido como telemetría y es un gran aliado de la telemedicina, cuyo objetivo es establecer la comunicación de un paciente con un profesional de la salud, brindándole herramientas de diagnóstico, monitoreo y tratamiento en pro de mantener su salud y mejorar su calidad de vida.
Sin embargo, la telemedicina afronta los desafíos que conlleva gestionar la información que se relaciona directamente con los pacientes, comenzando por atender una población extremadamente numerosa, con fuentes variables de información y con la necesidad de disponerla lo suficientemente rápido donde se requiera, con el objetivo de brindar un servicio eficiente y garantizar su trazabilidad.

Actualmente la empresa Integradores de Información - Vitalbox SAS es una de las pioneras en telemetría. Uno de sus objetivos es centralizar la información del paciente relacionada al área de la salud, vinculando la historia clínica, el perfil de vida y sus signos vitales de forma segura y confiable para disponerla a su EPS, IPS o el profesional de la salud correspondiente cuando sea necesario. Esto permitirá diagnosticar, tratar o monitorear el estado de salud de un paciente de forma local o remota, facilitando el servicio de atención médica para tomar cualquier tipo de decisión de manera rápida y eficaz.

Sin embargo, ante la necesidad de digitalización impuesta por la pandemia se ha incrementado el volumen de información que deben integrar a su repositorio, estimando tandas del orden de los cientos de Giga Bytes (GB) de información por día con posibilidad de incrementarse según la demanda. La información puede provenir

de diferentes tipos de archivos de texto e imágenes, cuyo procesamiento (particularmente para los pdf y los diferentes formatos de imagen) puede tardar un tiempo considerable bajo metodologías tradicionales que consisten en realizar múltiples transformaciones del archivo hasta llevarlo a un dominio textual, lo que puede bloquear el hilo principal del sistema, y por ende interrumpir la prestación de los servicios de la plataforma. Motivo por el cual se propone implementar técnicas de procesamiento multinúcleo para mantener de forma independiente el servicio y el procesamiento; y adicionalmente, se propone implementar técnicas de procesamiento de imágenes que permitan mapear las zonas donde se encuentra la información de interés y extraerla aplicando algoritmos de OCR (Optical Character Recognition) para evitar el procesamiento de la totalidad de cada archivo, disminuyendo a su vez el tiempo y el costo computacional, cuyo beneficio escalará proporcionalmente con el volumen de la información, y de esta forma optimizar el flujo de integración masiva de información al repositorio.

Objetivos

Objetivo General:

Optimizar el flujo de carga masiva de información proveniente de archivos tipo pdf e imagen en la plataforma MyVitalbox mediante técnicas de procesamiento multinúcleo y OCR con miras a mejorar la escalabilidad y eficiencia de un servicio que es favorable para la telemedicina.

Objetivos Específicos:

- Identificar los algoritmos óptimos de mapeo y reconocimiento de texto en imágenes y archivos pdf a partir de técnicas OCR que permitan extraer información relevante evadiendo los métodos tradicionales para optimizar el flujo de procesamiento de datos de la plataforma MyVitalbox.

- Evaluar eficacia y tiempo de ejecución competitivo de los algoritmos identificados, mediante su aplicación en un dataset de prueba de tamaño limitado, comparando resultados con la metodología de procesamiento tradicional para estimar la conveniencia de su integración.

- Implementar los algoritmos desarrollados en el servidor de pruebas de la plataforma MyVitalbox y someterlos a ejecución bajo condiciones habituales para validar su escalabilidad y verificar la optimización en la técnica de procesamiento.

- Adaptar técnicas de procesamiento multinúcleo a los algoritmos implementados en el servidor de pruebas para garantizar el procesamiento eficiente de la información sin la suspensión del hilo principal que sostiene la plataforma, con miras a mejorar la escalabilidad de los servicios y procesos relacionados con la telemedicina.

Marco teórico

La Telemedicina es un término incorporado desde la década de los 70s, haciendo referencia a la prestación de servicios médicos a distancia. Actualmente, implica el uso de las tecnologías de la información y la comunicación (TIC) con el propósito principal de proveer soporte clínico y superar las barreras geográficas para conectar usuarios que no se encuentran en la misma ubicación física, mejorando así, los resultados en el área de la salud al posibilitar el intercambio de información valiosa para tratar y prevenir enfermedades, además de investigar, educar y evaluar con miras al crecimiento del sistema de salud, tanto a nivel individual como de comunidad **[1]**.

Al ser una ciencia en constante crecimiento y evolución, debe incorporar la tecnología de vanguardia para adaptarse a las necesidades de la sociedad, que actualmente emergen de un ecosistema tecnológico que incluye el uso de computadores, centrales de cómputo, nubes de datos, aplicaciones, infraestructuras, arquitecturas tecnológicas, redes de comunicación y múltiples herramientas de hardware, firmware y software cuyo propósito gira en torno a los datos; donde almacenar, proteger, recuperar y procesar son las principales tareas que se realizan con la intención de extraer información valiosa que facilite la toma de decisiones que decidan el futuro de una empresa, una comunidad, un mercado completo e incluso, el mundo **[2]**.

Dentro de estas tecnologías, han surgido diversas ramas especializadas para suplir las necesidades de las TIC, como el internet de las cosas (IoT), la inteligencia artificial (AI), la visión por computadora, la seguridad informática, las técnicas de análisis “Big Data”, Business Intelligence (BI) y algunas otras que suelen escucharse comúnmente dentro de este ecosistema que buscan transformar datos crudos en información; incrementando la demanda de perfiles tecnológicos con la capacidad de generar sistemas de adquisición de datos mediante el uso de sensores y microcontroladores, crear inmensos repositorios de datos (Data Warehouse) que permitan registrar el estado de un negocio en cada instante de tiempo, generar modelos predictivos de un fenómeno al extraer, transformar y cargar información masiva, entre otros; usando para todo esto diversos algoritmos especializados de procesamiento de datos, ya sean texto, imágenes, videos, audios, y los diversos formatos en los que puedan encontrarse estas fuentes **[3]**.

Como puede apreciarse, el núcleo de las TIC son los datos, los cuales con el paso del tiempo pueden volverse inmanejables cuando provienen de fuentes variables en grandes volúmenes y requieren un tratamiento de alta velocidad; este problema de las “3 V (variable, voluminoso, veloz)” es a lo que se denomina en este ecosistema como Big Data **[4]** y es el punto principal de este proyecto, cuyo objetivo es optimizar el flujo de procesamiento de datos para evitar que se conviertan en un problema a medida que escalan en complejidad, y que bajo el contexto tratado consiste en técnicas de procesamiento de imágenes, aplicando particularmente algoritmos de reconocimiento de caracteres, conocidos como OCR por sus siglas en inglés (Optical Character Recognition) y pueden o no, incluir características de inteligencia artificial para afrontar desafíos propios de la fuente que se presentan en la tabla 1 **[5]**.

Tabla 1. Desafíos en OCR.

Propiedad del texto en la imagen	**Desafíos**

Densidad	Unificado, disperso, expandido o contraído
Estructura	Estructurado en filas y columnas, o disperso espacialmente y rotado
Fuente	Escrito a mano, impreso, fotografiado o escaneado
Tipo de caracter	Símbolos y formas válidas de acuerdo con el lenguaje
Artefactos	Dependiente de la calidad visual del texto, puede ser limpio y claro o borroso y amorfo
Localización	Ubicación estándar o aleatoria en la imagen

Finalmente, una herramienta adicional que es usada comúnmente en problemas de Big Data es el procesamiento multinúcleo, que consiste en una técnica de paralelizar tareas usando los diferentes cores que puede tener una CPU para generar procesos independientes en recursos computacionales. Sin embargo, suele confundirse con el concepto de multi-hilos o multi-threading, que permite ejecutar múltiples tareas en background dentro del mismo core, aunque de forma secuencial tal y como se evidencia en la figura 1 **[6-8]**.

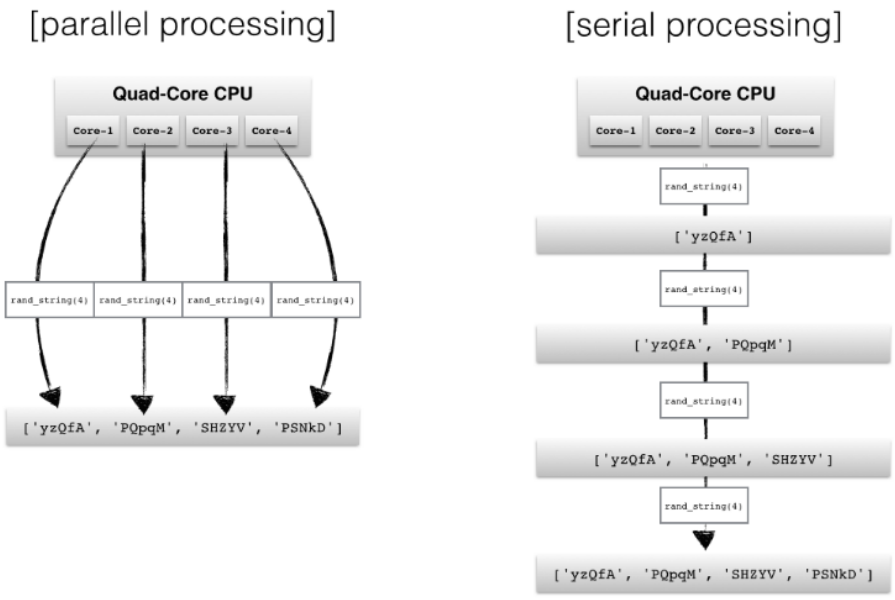

Figura 1. *Diferencia entre los esquemas de ejecución de tareas con procesamiento multinúcleo (paralelo) y multihilo (secuencial).*

Metodología

Para el cumplimiento de los objetivos del proyecto, es necesario realizar una serie de actividades que se dividen en 5 etapas: Investigación, desarrollo, implementación, rendimiento y conclusión. Estas se describen a continuación con los detalles que involucran cada una de ellas.

Etapa 1. Investigación:

En esta etapa inicial debe realizarse una revisión contextual y técnica de las tecnologías involucradas en el proyecto, su estado del arte, técnicas, algoritmos y frameworks disponibles para el desarrollo e implementación de los requerimientos del proyecto.

1. **Big Data:**
   Revisión de los casos de consideración de Big Data, sus causas y técnicas empleadas para afrontar estas situaciones, así como los lenguajes de programación y sus bibliotecas eficaces para desarrollar e implementar este tipo de soluciones.

2. **Procesamiento de imágenes:**
   Estudio de las técnicas de transformación y manipulación de imágenes que permitan acondicionar las fuentes de información para su posterior análisis y procesamiento con algoritmos de OCR.

3. **Procesamiento Multinúcleo:**
   Investigación de frameworks y bibliotecas aptos para paralelizar la ejecución de tareas mediante el uso de los distintos núcleos de la CPU.

Etapa 2. Desarrollo:
Una vez conocido el estado del arte y las herramientas óptimas para el desarrollo del proyecto, se procede con el desarrollo de algoritmos y pruebas de los mismos, con un dataset de tamaño limitado.

1. **Procesamiento de Imágenes:**
   Este es el algoritmo inicial que debe aplicarse a las fuentes de información según su calidad cuando sea necesario con el objetivo de facilitar el reconocimiento de caracteres, manteniendo una complejidad baja de procesamiento para minimizar el tiempo de ejecución y garantizar la escalabilidad de su implementación.

2. **Reconocimiento de caracteres:**
   Consiste en el desarrollo del algoritmo principal que debe mapear las zonas de información relevante en telemedicina, e identificar y descifrar el texto contenido para su posterior tratamiento y almacenamiento.

3. **Almacenamiento:**
   Luego de extraer la información de interés, debe desarrollarse un algoritmo de almacenamiento en la base de datos, utilizando el driver disponible en el lenguaje o framework en el cual se haya desarrollado la técnica de OCR.

4. **Procesamiento multinúcleo:**

Para finalizar con la etapa de desarrollo, solo queda adaptar las técnicas de paralelización de tareas identificadas a los algoritmos generados.

5. **Pruebas:**
   Esta es una actividad que se desarrolla iterativamente y en paralelo con el desarrollo de los algoritmos para probar de forma individual su funcionamiento con fuentes de datos como archivos individuales. Además, se determina el dataset de tamaño limitado que se va a emplear y se pone a prueba el flujo de procesamiento desarrollado.

Etapa 3: Implementación:
Posterior al desarrollo y prueba de los algoritmos, puede iniciarse la implementación para realizar validaciones bajo condiciones habituales.

1. **Backend:**
   Dependiendo del lenguaje en el que se hayan desarrollado los algoritmos, debe revisarse la forma de integrarlos con la plataforma, ya sea directamente en la API o mediante técnicas de multiprocesamiento.

2. **Integración en servidor de pruebas:**
   Luego de la implementación, sólo queda realizar la integración al servidor de pruebas de la plataforma para someter el módulo de procesamiento diseñado a un flujo de datos habitual.

Etapa 4: Análisis de Rendimiento:
Esta etapa busca analizar aspectos críticos del flujo de procesamiento implementado, con los cuales debe poderse comparar con la metodología tradicional que tiene la plataforma.

1. **Eficacia:**
   Debe determinarse si el algoritmo OCR implementado se adapta a la plataforma y realiza el mapeo de las zonas de información relevante e identifica el texto contenido para su tratamiento y almacenamiento tal como se esperaba inicialmente, tanto para volúmenes grandes como pequeños.

2. **Velocidad:**
   Cuando se avalen funcionalmente los algoritmos diseñados, deben implementarse métricas de tiempo de ejecución que permitan analizar la velocidad de procesamiento por archivo o por lotes identificando el tipo de fuente; es decir, si corresponde a imagen o pdf.

3. **Escalabilidad:**
   Debe validarse la capacidad que tiene el flujo de procesamiento, de escalar a grandes volúmenes de información sin perder eficacia ni aumentar considerablemente el tiempo de ejecución.

4. **Independencia:**
   Aquí se determina si el procesamiento se lleva a cabo sin usar recursos del proceso principal que mantiene los servicios de la plataforma, y si se garantiza su continuidad bajo eventos inesperados del servidor sobre la plataforma.

Etapa 5: Conclusión:
Finalmente, se estudian la metodología tradicional y las técnicas de procesamiento implementadas para corroborar la hipótesis inicial sobre la optimización del flujo de procesamiento, en pro de los servicios de telemedicina que se favorecen de la plataforma de MyVitalbox.

1. **Comparativa con la metodología tradicional:**
   Finalmente se realizan las comparaciones pertinentes con la metodología tradicional y el flujo de procesamiento implementado bajo el análisis de rendimiento realizado, aquí se confronta la eficacia, el tiempo de ejecución y la escalabilidad de los resultados.

2. **Viabilidad:**
   Para culminar, se define la viabilidad de implementación del proyecto en la plataforma MyVitalbox para cumplir con los objetivos establecidos basado en el análisis de rendimiento y la comparativa con la metodología tradicional.

En la figura 2, se presenta un diagrama que detalla las actividades a realizar para cumplir con los objetivos del proyecto.

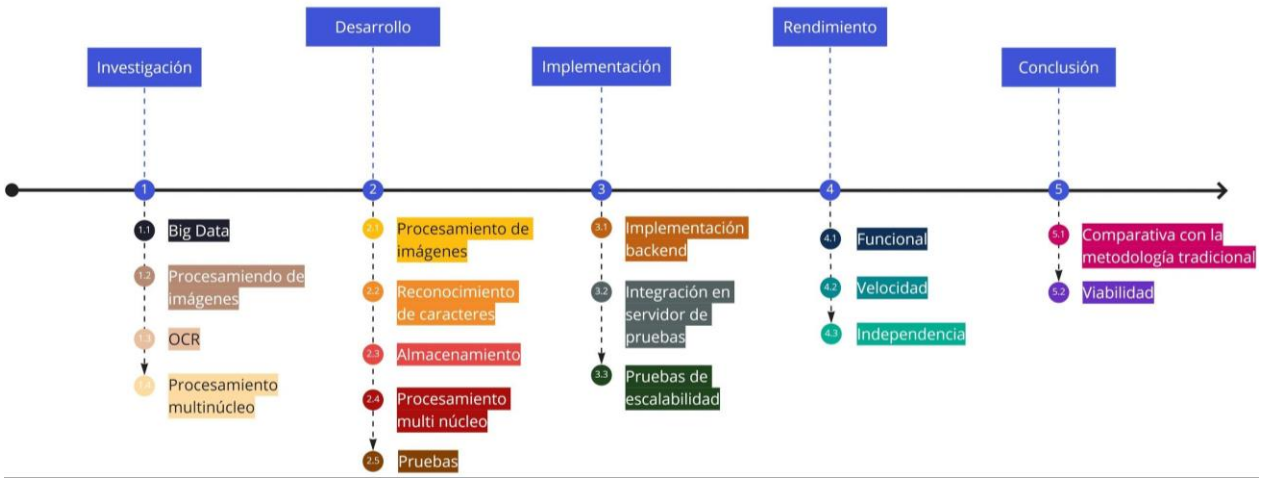

Figura 2. *Diagrama de etapas y actividades planteadas para la ejecución del proyecto.*

Resultados y análisis

1. **Investigación**

   El núcleo de los negocios, productos y servicios que son basados en software, son los datos, los cuales a partir de su entendimiento y organización generan

conocimientos del comportamiento de un fenómeno, permitiendo tomar decisiones que produzcan ventajas y beneficios, motivo por el cual han surgido tendencias que promueven la inversión en la ingeniería aplicada a la ciencia de los datos, incluyendo herramientas o metodologías de registro, almacenamiento, generación de métricas estadísticas y modelos predictivos.

Sin embargo, con el paso del tiempo la acumulación y el flujo masivo de datos que son el insumo principal de estas tecnologías, se da origen a conceptos problemáticos como es el caso del Big Data, el cual se genera técnicamente cuando la memoria RAM no tiene la capacidad de procesar tres componentes de los datos: el volumen, la velocidad y su variabilidad en un propósito definido. Las soluciones modernas a este tipo de problemas implican:

- Mejorar el hardware del dispositivo, aunque suele tener un costo elevado y no solucionan el problema por completo, dado que por un lado depende en principio de la capacidad de la tarjeta madre para aumentar la memoria RAM, y por otro lado los datos pueden superar aún la nueva capacidad bajo alguna circunstancia.

- Integrar software de terceros como SAS, IBM, Amazon y otros que gestionan el tratamiento de los datos por sí mismos y brindan herramientas de análisis que facilitan el proceso, convirtiendo el problema en uno de integración que puede ser más fácil de manejar. Sin embargo, puede considerarse como una posible desventaja el costo del servicio.

- Uso de herramientas especializadas en el tratamiento de datos, como el paquete de Anconada con Python y R optimizado para ciencia de datos, o Hadoop y Spark cuya especialidad mejora el rendimiento de múltiples procesamientos de datos masivos dada su optimización para temas de Big Data **[9]**.

- Optimizar los servicios de bases de datos en función de las características de los datos. Esta solución involucra la estructuración adecuada para el modelo de negocio y la selección óptima del motor de almacenamiento de acuerdo con el diseño, teniendo en cuenta si el flujo será de escritura o lectura en su mayoría, la frecuencia con la que se presentan dichas transacciones, el servidor en el que se almacenan y el lenguaje en el que se procesan **[10]**.

- Optimizar el flujo de procesamiento haciendo uso de herramientas como los “generadores” que incorporan la mayoría de los lenguajes de programación, y actúan mediante la suspensión temporal del

procesamiento de un objeto recordando su último estado y dejando una marca de pendiente que puede ser ejecutada cuando se desee **[11]**.

Las soluciones expresadas previamente, no están orientadas a un tipo de dato en particular, pero dado que para este caso el núcleo de los datos son las imágenes, existen estrategias con enfoques particulares que pueden ser adaptadas para atacar este problema, involucrando técnicas de procesamiento de imágenes y procesamiento multi núcleo que permiten la paralelización de tareas.

Estas estrategias pueden ser implementadas de múltiples formas en función del lenguaje de programación a utilizar. Respecto al procesamiento de imágenes, en Nodejs que es la herramienta en la que está desarrollada la plataforma de MyVitalbox para la cual se diseña esta solución, se tienen disponibles bibliotecas como jimp, Sharp, image-js y algunas otras, sin embargo debido a que es un lenguaje que surge de un entorno de trabajo en el navegador su enfoque principal es la manipulación básica para la transmisión y visualización de imágenes, implementando técnicas de compresión, operaciones básicas de ajuste de tamaños, recortes, rotaciones, manipulación de color y filtrado básico **[12]**, por lo que no es óptimo para construir un flujo de procesamiento complejo que sea dinámico y versátil.

Por otro lado, en Python también existen varias bibliotecas para esta tarea, como son Pillow, Scikit-image, Scipy y otras, pero sin duda la más popular por su gran capacidad es OpenCV, la cual está especializada en aplicaciones de visión por computadora debido a la inmensa cantidad de algoritmos implementados que permiten realizar operaciones básicas, procesamiento de alta complejidad, detección de características, análisis de video, machine Learning, detección de objetos y muchas otras **[13]**, dentro de las cuales se encuentran funcionalidades de detección y reconocimiento de texto, más conocido como OCR y para las cuales se emplea el motor de Tesseract **[14]**.

OpenCV permite implementar técnicas de transformación y procesamiento que acondicionen las imágenes de insumo mediante el filtrado, la extracción y ecualización de color, la corrección del histograma mediante el factor gamma, la binarización simple y adaptada, o la binarización mediante algoritmos como el de Otsu, transformaciones morfológicas, identificación de contornos y generación de máscaras **[15]**.

Adicionalmente, al ser los archivos pdf uno de los principales insumos es necesario transformarlos a imágenes previamente para aplicar el flujo de procesamiento a desarrollar, para lo cual Python provee la posibilidad de conversión de archivos pdf a imágenes mediante módulos como pdf2image **[16]**.

Finalmente, respecto al procesamiento multinúcleo, Nodejs provee el módulo child-process que permite lanzar tantos subprocesos como la capacidad de los núcleos de la CPU lo permitan, adicionalmente son lanzados de forma asíncrona por defecto, evitando bloquear el ciclo de eventos del hilo principal en el que corre Nodo, siendo este uno de los motivos de su nombre **[17]**.

2. Desarrollo

Procesamiento de Imágenes

Para determinar el flujo de procesamiento óptimo se diseñó un módulo que integra gran parte de las funcionalidades disponibles en OpenCV para transformar y tratar una imagen estática o dinámica, el cual en conjunto con una interfaz diseñada en PyQt permiten procesar en tiempo real imágenes captadas por la cámara del computador e imágenes estáticas cargadas desde el disco duro. Adicionalmente se integra la funcionalidad de convertir archivos pdf a imagen incorporando el módulo pdf2image y las funcionalidades de detección y reconocimiento de texto que provee el motor de Tesseract mediante el módulo PyTesseract. A continuación se presentan imágenes de los componentes principales del módulo de procesamiento.

En la figura 3 se presentan las funcionalidades básicas y la parametrización del flujo de procesamiento, el cual incluye lo siguiente:

- Redimensión de imagen (zoom):
- Interpolación: De área, lineal y cúbica.
- Rotaciones
- Operaciones punto a punto
- Filtro o suavizado: Blur, Gaussian Blur y Median Blur
- Extracción de color: Escala de Gris o RGB (Rojo, Verde, Azul)
- Corrección Gamma: Para la optimización del histograma de la imagen.
- Binarización o umbralización: Simple, adaptada y algorítmica.
- Simple: binaria, binaria inversa, truncada, a cero, a cero inversa.
- Adaptada: De Promedio y Gaussiana con binarización normal e inversa.
- Algorítmica: Algoritmo de Otsu con binarización normal e inversa
- Transformaciones morfológicas: Erosión, dilatación, apertura y cierre con iteraciones múltiples
- Detección de Bordes mediante el algoritmo de Canny
- Identificación de contornos
- Generación y aplicación de máscaras

Figura 3. *Módulo de parametrización de técnicas de procesamiento de imágenes que serán aplicadas en cascada para obtener la transformación deseada de la imagen objetivo.*

En la figura 4 se presenta una demostración de filtrado, extracción de color, y umbralización adaptada para una imagen de prueba.

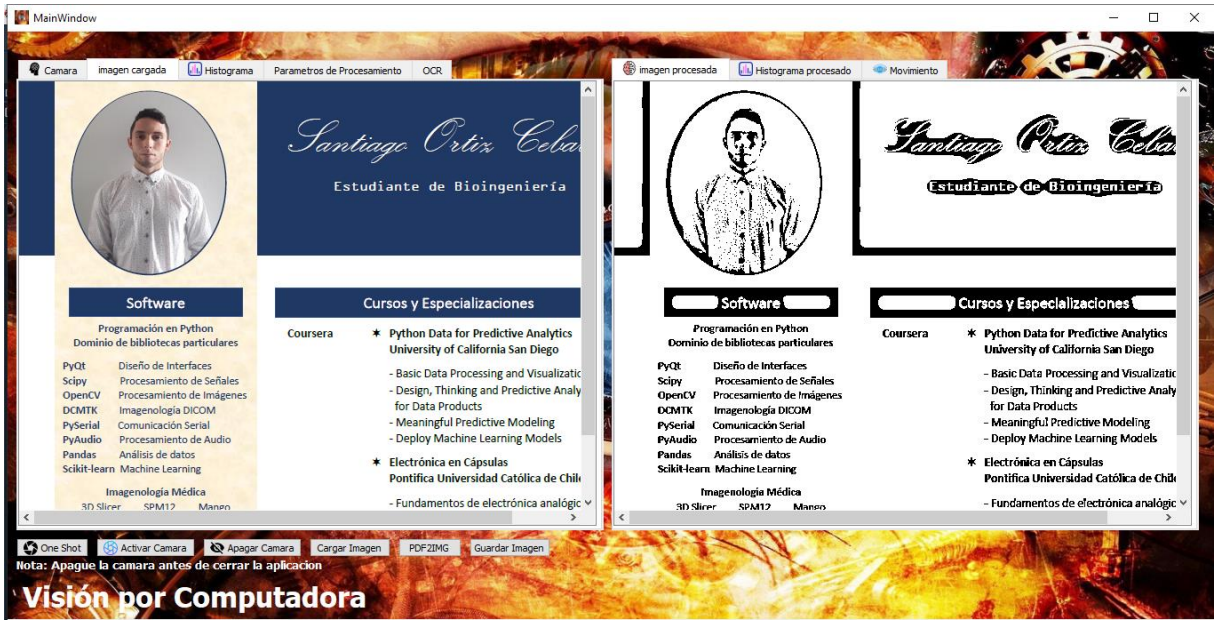

Figura 4. *Demostración del flujo de procesamiento para una imagen estática.*

Reconocimiento de caracteres

El módulo de pruebas desarrollado integra las siguientes funcionalidades del motor de Tesseract:

- Detección de orientación de texto
- Detección de caracteres
- Detección de palabras
- Detección de lenguaje

A continuación se presenta una demostración con una orden médica de prueba tomada de un pdf y transformada en imagen mediante el módulo pdf2image. En la figura 5 se muestra el texto identificado por el motor de OCR (procesado y formateado) y la imagen resultante con las palabras detectadas encerradas por un rectángulo de color rojo.

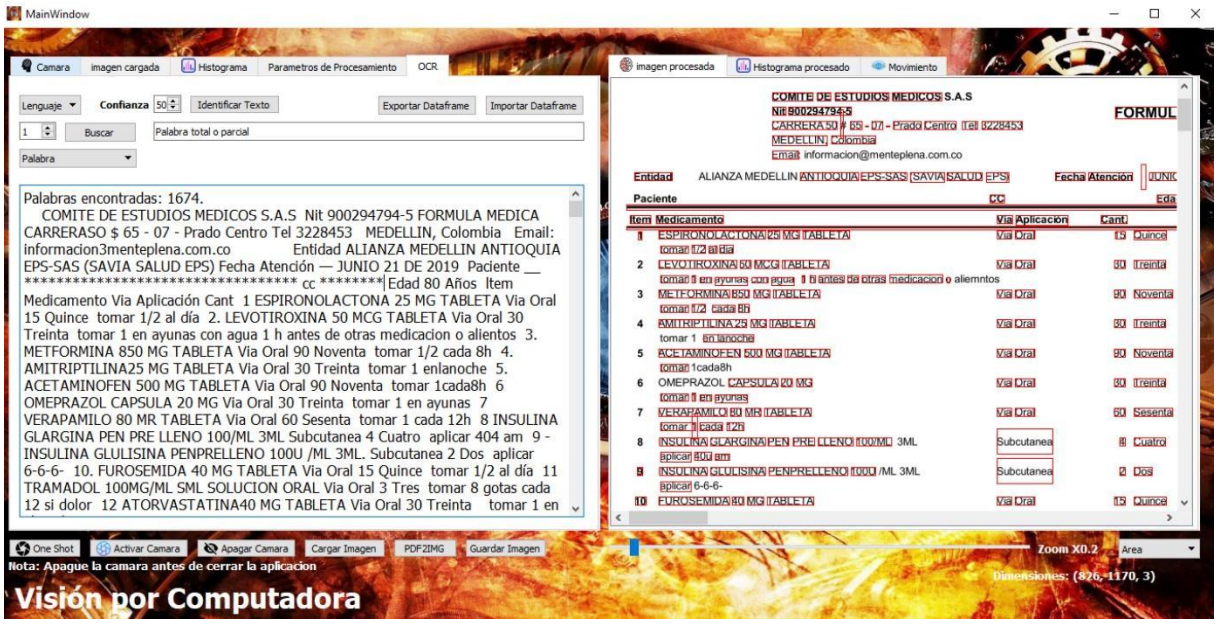

Figura 5. *Texto identificado en una orden médica de prueba tomada de un pdf (La identificación del paciente ha sido censurada).*

Extracción de información

Dado el formato del texto resultante del motor de Tesseract, se da la necesidad de hacer una implementación no planeada de un algoritmo de extracción de información mediante el uso de expresiones regulares. En la figura 6 se presenta una demostración de su funcionamiento.

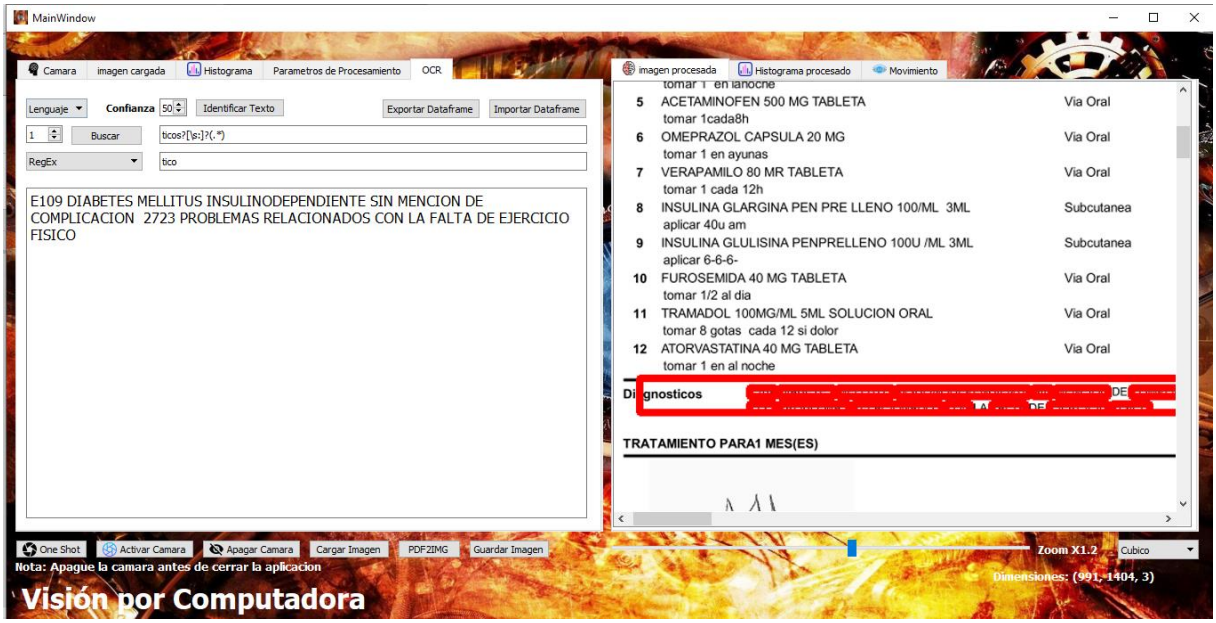

Figura 6. *Extracción del diagnóstico del paciente mediante expresiones regulares Luego de aplicar técnicas OCR.*

Almacenamiento

Para el almacenamiento de la información de interés (predefinida) extraída del algoritmo de OCR y procesada con expresiones regulares, se generó una clase denominada MySQL_Engine que permite mediante el driver de MySQL para Python (mysql.connector) entablar comunicación con una base de datos relacional e ingresar registros dinámicamente en las tablas deseadas.

Procesamiento multinúcleo

Usando el módulo de subprocesos que provee Nodejs (child-process) se generó un Script que permite ejecutar el módulo de Python que integra en un entorno virtual la conversión pdf2image, procesamiento de imágenes, OCR e integración con MySQL Server. Implementando la comunicación entre procesos y la atención a eventos de ejecución y levantamiento de errores para garantizar su sostenibilidad, además de la desvinculación de procesos para prevenir al “padre” esperar a que los subprocesos finalicen, así como la continuidad de los procesos “hijo” aunque el padre se detenga o finalice por algún motivo con el fin de garantizar la independencia del hilo principal, evitando el bloqueo funcional de la plataforma **[17]**.

En la figura 7 se presenta un segmento del script de prueba que se ejecuta desde Nodejs y el subproceso de Python ejecutado en segundo plano y en paralelo, el cual recibe los parámetros requeridos mediante variables de entorno y se comunica con el Backend de la plataforma mediante la librería “sys” de Python y el manejo de eventos que provee Nodejs.

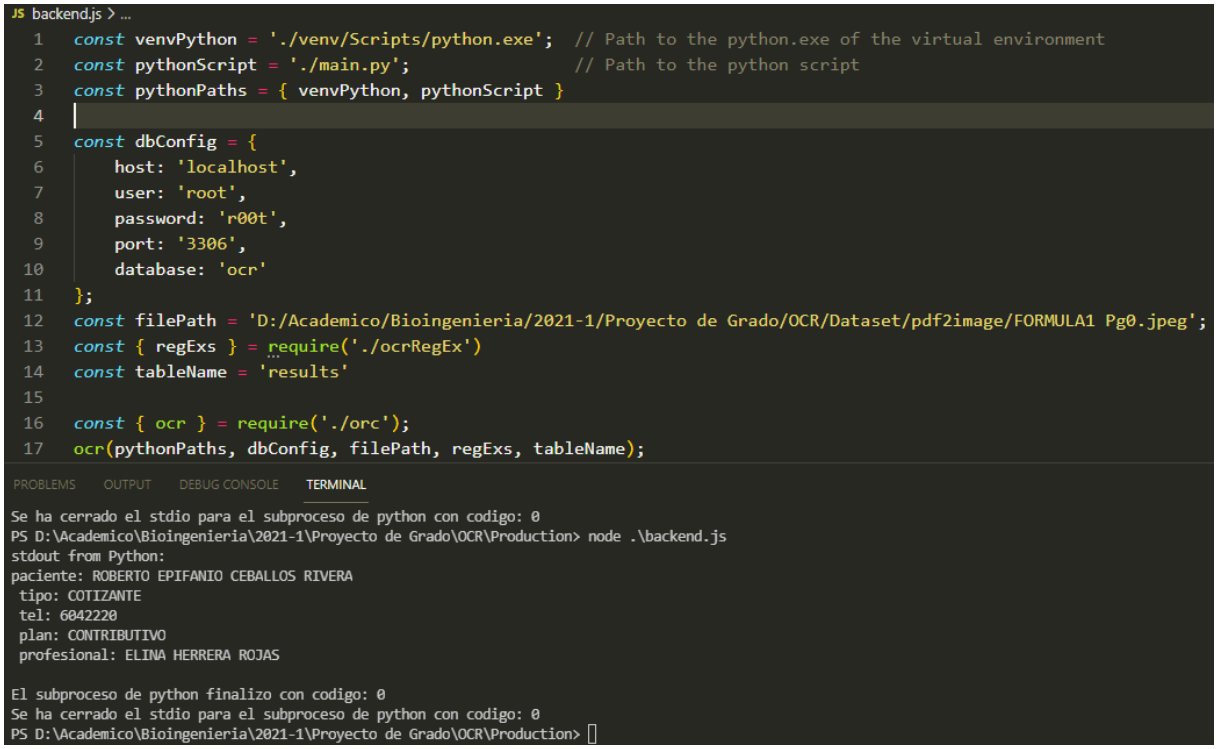

Figura 7. *Ejecución del Script de Python como subproceso de Nodejs. En la terminal que aparece al inferior de la imagen se aprecia el resultado del procesamiento realizado por el script en Python.*

El Script presenta los parámetros enviados a Python, correspondientes a las credenciales de conexión a la base de datos, la ruta a la imagen objetivo y las expresiones regulares prediseñadas para la extracción de la información. En la parte inferior de la imagen se evidencia la terminal de ejecución con el resultado de la información extraída por el módulo de OCR reportado desde Python.

Pruebas

Las pruebas se realizaron en un dataset de tamaño limitado que incluye órdenes de medicamentos, formulaciones médicas e Historias Clínicas. En la tabla 2 se presenta un resumen de los resultados obtenidos.

Tabla 2. Resumen de resultados en pruebas con dataset de tamaño limitado.

	A	B	C	D	E	F	G	H
1	**Resultados del Dataset de Prueba**							
2	**Archivo**	**Formato**	**Procesamiento**	**Tamaño**	**RegEx Usadas**		**Resultado**	**Observaciones**
3	FM Diabetes pg0	JPEG y PNG	**Procesamiento 1** -Extracción de color -Otsu + Binary - Zoom out (x0.5) + Interpolador de área **Procesamiento 2** -Extracción de color -Otsu + Binary - Zoom in (x1.2) + Interpolador cúbico	-Original: 4132x5848 -Resultante 1: 2066x2924 - Resultante 2: 4958x7018	**- Paciente / Documento / Edad** paciente[:\s]*(.*)cc[:\s]*(\d+)[.\s]*edad[:\s]*(\d+) **- Entidad** entidad[\s:]*([\w\s-]+)\(([\w\s]+)\) **- NIT** nit[\s:]?([\d\.-]+) **- Email** mail[\s:\.]*([-\w\._]+)	**- Fecha atención** keyword: \d \s?(\w+\s?\d+\s?de\s\d{4,4}) **- Medicamentos** ([\w]+)\s\d*\s?[/|\s]?m[c|g|\r] \s\d?\s([\w\s]+)\s\d*\s?[/|\s]?m[c|g|\r] **- Diagnóstico** diagn[oó]sticos?[\s:]?(.*) **- Problemas** problemas[\s:]?([\w\s]+)	8/10	No extrae medicamentos de más de 1 palabra o con detalles de uso
4	FORMULA 1	JPEG	**Procesamiento 1** - Zoom out (x0.4) - Interpolador de área	-Original: 4250x5500 -Resultante:	**-Fecha de atención** \d{4,4}-\d{2,2}-\d{2,2} **-Código de Medicamentos** Búsqueda por palabra: MD **-Sede** sede:?\s?([\w\s]+)\sPacien **-Paciente** Paciente:?\s?([\w\s]+)	-**Contrato** Contrato:?\s?([\w\s]+)(?=\splan|tipo) **-Tipo de Usuario** usuario:?\s?(\w+) **-Teléfono** Tel[eé]fono:?\s?([\d-]+) **-Plan** plan:?\s?(\w+) **-Profesional** profesional:?\s?([\w\s]+)\s	10/10	
5	HC y Orden Monitoreo Amb Pres Art Sist Pg0	JPEG	**Procesamiento 1** -Extracción de color -Otsu + Binary - Zoom out (x0.4) + Interpolador de área	-Original: 4250x5500 -Resultante: 1700x2200	-**CC** cc\s?:?-?(\d+) **-Servicio** servicio.?.?([\w\s]+)\s| **-Entidad** entidad.?([\w\s\.]+)	**-Fecha de atención** \d{4,4}-\d{2,2}-\d{2,2} **-Farmacológicos** farmacol[oó]gicos:?\s?.*	9/10	La información entre paréntesis no se extrae a la perfección debido a que son un caracter reservado del módulo de RegEx
6	Orden Colonoscopia Total Pg0	JPEG	**Procesamiento 1** -Zoom Out (x0.5) - Defecto PyTesseract	-Original: 4250x5500 -Resultante: 1375x2067	-**EPS** eps.?(\w+) -**NIT** nit.?\s?(\d+) -**Solicitud** solicitud\sde\s(.*) -**Nro Aprobación** apro[vb]aci[oó]n:?\s?(\d+) -**Fecha atención** \d{2,2}/\d{2,2}/\d{4,4} -**Nombre paciente** nombre:?\s?([\w\s]+) -**Identificación paciente** identificaci[oó]n:?\s?(\w{2,2}\s?\d+)	-**Sexo** sexo:?\s?(\w+) -**Edad** edad:?\s?(\d+) -**Contrato** contrato:?\s?([\w\s\.:]+\s[\d-]+) -**Tipo usuario** usuario:?\s?(\w+) -**Procedimiento** \s{2}\d+\s-\s([\w\s]+) key: - -**Cédula del médico** (\w{2,2}\s\d+)	7/10	-Los diagnósticos de EPS Sanitas están codificados. De conocerse exactamente el formato es posible recuperarlos por completo - El procedimiento al estar embebido en una tabla no es preciso, sin embargo es posible recuperarlo

3. Implementación

La implementación se realizó mediante el Framework de Loopback en el cual se basa el **Backend** de la plataforma de MyVitalbox para el desarrollo de APIs en Nodejs. En el cual se expone un servicio **REST** tipo **POST** en el "endpoint" **/Python/ocr** que puede ser usado en el servidor de pruebas, tanto de forma remota como desde el interior de la aplicación mediante el método **ocr** del modelo **Python**.

Este servicio recibe como argumentos la ruta local al archivo objetivo, los atributos o campos que serán registrados en la base de datos y las expresiones regulares a partir de las cuales extraerá la información deseada del archivo objetivo. Una vez se validan los parámetros ingresados, se levanta en segundo plano un subproceso en Python que ejecuta el módulo de OCR diseñado para el procesamiento y almacenamiento de la información. A continuación se presenta un ejemplo de la trama de argumentos que recibe el servicio implementado:

Método remoto:

Encabezado:

Content-type: application/json

Accept: application/json

Cuerpo:

{
"rutaImagen":"D:/Academico/Bioingenieria/2021-1/Proyecto de Grado /OCR/Dataset/pdf2image/FORMULA1 Pg0.jpeg",
"regExs": {
"sede": "sede:?\\s?([\\w\\s]+)\\sPacien",
"paciente": "Paciente:?\\s?([\\w\\s]+)\\sID",
"contrato": "Contrato:?\\s?([\\w\\s]+)(?=\\splan|tipo)",
"tipo": "usuario:?\\s?(\\w+)",
"tel": "Tel[eé]fono:?\\s?([\\d-]+)",
"plan": "plan:?\\s?(\\w+)",
"profesional": "profesional:?\\s?([\\w\\s]+)\\s"
},
"atributos": "sede, paciente, contrato, tipo, telefono, plan, profesional"
}

Método del modelo para su uso desde el interior de la API:

El modelo de Python y su método de OCR recibe como parámetro la ruta de la imagen al igual que el método remoto, sin embargo, las expresiones regulares y los atributos o campos que serán registrados en base de datos, pueden ser recuperados adicionalmente, de un objeto importado desde un archivo de JavaScript, o quemados al interior de la función para facilitar su ejecución y brindarle un comportamiento versátil que le permita ajustarse a requerimientos futuros.

4. Análisis de rendimiento

Una vez implementado el servicio y realizadas las evaluaciones funcionales usando el dataset de prueba se evidencian los resultados que se presentan a continuación.

Eficacia

Con base en los resultados consignados en la tabla 2, y la columna de resultados obtenidos (medidos de forma cualitativa con base en la cantidad y

calidad de la información extraída) se tiene una eficacia del 85% calculada de la forma:

$$E\% = \frac{resultado\ obtenido}{resultado\ esperado}\ x\ 100 = \frac{34}{40}\ x\ 100 = 85\%$$

Velocidad

La velocidad medida en tiempo de ejecución que toma extraer y almacenar la información deseada de un archivo pdf al cual se le aplica un procesamiento estándar de transformación de pdf a imagen, extracción de color, “Zoom in” con interpolación cúbica y binarización mediante el algoritmo de Otsu, a través de la API utilizando el método remoto, es de 5 a 6 segundos aproximadamente tal y como se muestra en la figura 8.

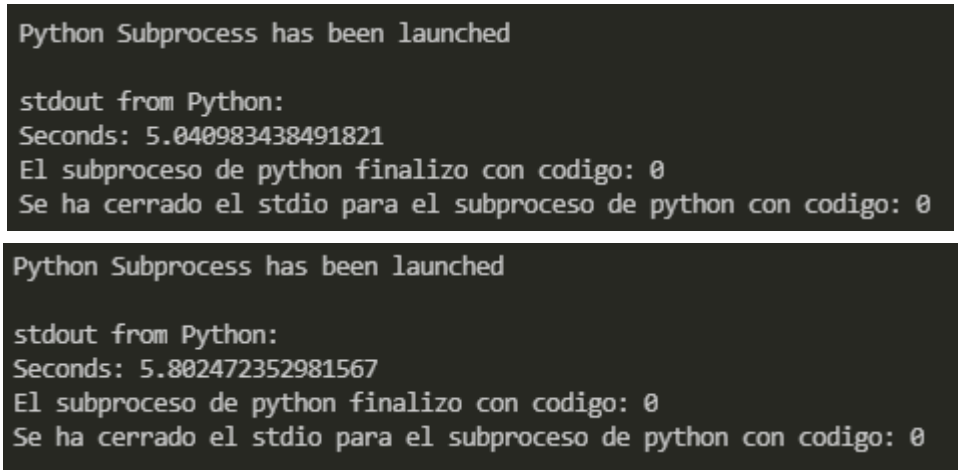

Figura 8. *Tiempo que toma extraer y almacenar información de un archivo pdf, convertido en imagen y transformado por un procesamiento de imágenes básico.*

Escalabilidad

La escalabilidad del servicio fue medida a través de dos indicadores, el primero siendo el tiempo de ejecución que tarda el módulo de OCR en procesar y almacenar la información extraída de múltiples archivos, y el segundo siendo la cantidad de subprocesos que pueden ser ejecutados de forma paralela.

En la figura 9 se presenta el resultado obtenido para el levantamiento de 4 subprocesos y en la figura 10 se presenta el resultado obtenido para el levantamiento de 7 subprocesos.

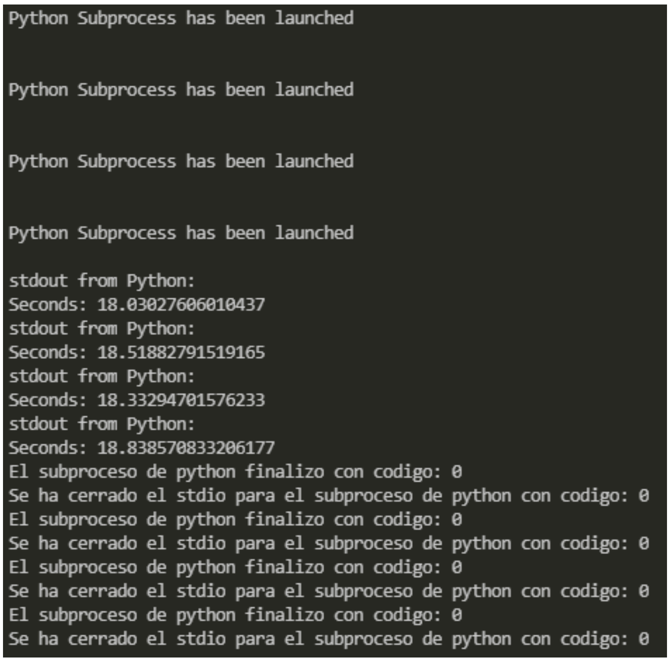

Figura 9. *Escalabilidad medida para 4 subprocesos.*

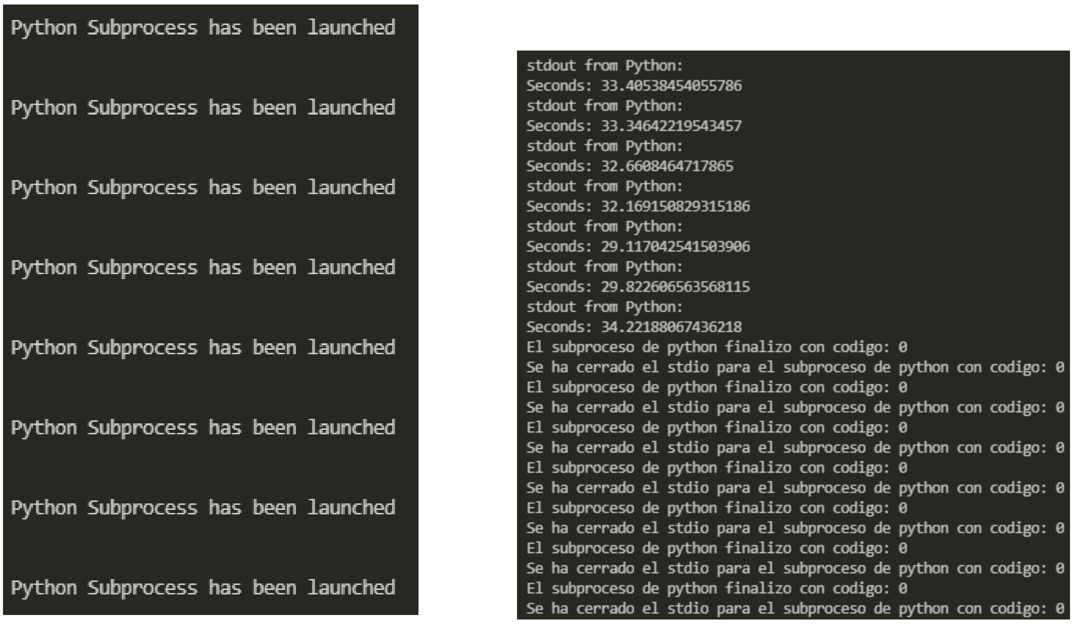

Figura 10. *Escalabilidad medida para 7 subprocesos.*

Independencia

La independencia se probó de dos formas diferentes, la primera la continuación de un proceso hijo aunque la API principal se detuviera, y la segunda validando

que la API permita levantar un proceso hijo y el rendimiento de otras peticiones continuas se mantuviera en buen estado. Para el primer caso fue inducido un error en un conector de datos en el framework de la API luego de ejecutar el método remoto /Python/ocr que levantara un subproceso, para cual se evidencia persistencia en el procesamiento debido a la información extraída y almacenada visualizada en la base de datos. Por otro lado, para el segundo escenario se ejecutó la misma petición para ejecutar el método remoto /Python/ocr múltiples veces, seguido de una petición aleatoria a cualquier servicio de la API, a lo cual se obtuvo como resultado una respuesta inmediata del servicio sin necesidad de esperar a que los subprocesos finalizaran.

Análisis de Resultados

Con base en los resultados obtenidos, el módulo diseñado permite extraer información de archivos tipo imágenes y pdf con un eficacia relativamente buena, los problemas principales radican en la extracción de información que se encuentre entre paréntesis, y aquella que se encuentre estructurada en tablas, en especial de forma vertical; esto debido a que el motor de procesamiento estructura de una forma diferente la información resultante de estos formatos, y aunque logre identificar los caracteres, se complica la extracción de información mediante la técnica empleada de expresiones regulares.

Por otro lado, en cuanto al rendimiento en velocidad y escalabilidad, no se obtienen resultados ideales al tomar 5 segundos procesar y extraer la información de un archivo genérico, y aunque es un tiempo que puede variar en función del tamaño original y la calidad del archivo (lo cual permite optimizar el flujo de procesamiento implementado), la escala de tiempo mínima proveniente del motor de Tesseract es de segundos, lo que es una unidad que no es escalable a grandes volúmenes de información, tal y como deja en evidencia los resultados presentados en las figuras 9 y 10, para las cuales se llega a 30 segundos con tan sólo 7 archivos, obteniendo una función de comportamiento aproximadamente exponencial para el tiempo.

Sin embargo, la independencia de ejecución del hilo principal de la API y de los procesos hijos, le brindan un gran potencial que puede ser aprovechado de otras formas tomando en cuenta la eficacia del módulo, en particular en comparación con la metodología tradicional que consiste en extraer del archivo el texto plano y almacenar tanto la información, como el archivo, consumiendo recursos que pueden ser excesivos y de poca utilidad a corto y mediano plazo; para lo cual el módulo diseñado ofrece una alternativa al proceso, extrayendo y almacenando únicamente la

información relevante, optimizando de esta manera, la gestión de la información que transite en estos formatos.

Conclusión

De acuerdo con el objetivo planteado, a los resultados obtenidos y los análisis realizados, se concluye que no es posible aplicar el módulo diseñado para cargar y procesar información de forma masiva debido al tiempo mínimo que toma procesar un archivo genérico utilizando el motor de Tesseract. Sin embargo, el módulo desarrollado para extraer y almacenar información precisa de imágenes y archivos pdf, presenta una eficacia lo suficientemente buena como para optimizar de una forma diferente a la planteada el proceso que lleva la información en la plataforma. Lo que debe tenerse en cuenta, es que el flujo de procesamiento aplicado para el análisis fue genérico y las posibles combinaciones de los parámetros de procesamiento pueden variar en función del archivo y el formato en particular que se desee tratar, lo cual es un factor que puede ser remediado mediante un modelo de inteligencia artificial (IA) que pueda ser entrenado para determinar los parámetros que deben ser aplicados de acuerdo a la calidad y el formato del archivo objetivo, sin embargo es un tema que se deja planteado para implementar mejoras a futuro ya que no es algo contemplado dentro del alcance del proyecto.

Adicionalmente la técnica empleada para la extracción de información puede ser complementada, dado que para el alcance del proyecto la problemática fue abordada mediante expresiones regulares, cuya eficacia es buena en la mayoría de escenarios, pero el formato resultante del motor de OCR también puede ser procesado por otras técnicas como *PLN* (procesamiento del lenguaje natural), el cual puede ser una alternativa o complemento para incrementar su versatilidad en función del formato o la situación en la que se desee aplicar este módulo.

Referencias bibliográficas

[1] Who.int. 2021. Telemedicine, Opportunities and developments in Member States. [online] Available at: <https://www.who.int/goe/publications/goe_telemedicine_2010.pdf> [Accessed 29 March 2021].

[2] Castagna, R. and Bigelow, S., 2021. What is Information Technology (IT)? – Definition from WhatIs.com. [online] SearchDataCenter. Available at: <https://searchdatacenter.techtarget.com/definition/IT> [Accessed 29 March 2021].

[3] Nair, D., 2021. The Evolution of Analytics with Data. [online] Medium. Available at: <https://towardsdatascience.com/the-evolution-of-analytics-with-data-8b9908deadd7> [Accessed 29 March 2021].

[4] Fedak, V., 2021. 10 hot trends of Big Data Analytics for 2017. [online] Medium. Available at: <https://towardsdatascience.com/10-hot-trends-of-big-data-analytics-for-2017-857679364890> [Accessed 29 March 2021].

[5] Rosebrock, A., 2021. OpenCV OCR and text recognition with Tesseract - PyImageSearch. [online] PyImageSearch. Available at: <https://www.pyimagesearch.com/2018/09/17/opencv-ocr-and-text-recognition-with-tesseract/> [Accessed 29 March 2021].
[6] GeeksforGeeks. 2021. Difference Between Multithreading vs Multiprocessing in Python - GeeksforGeeks. [online] Available at: <https://www.geeksforgeeks.org/difference-between-multithreading-vs-multiprocessing-in-python/> [Accessed 29 March 2021].
[7] W. and Franco, G., 2021. What are the differences between the threading and multiprocessing modules?. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/18114285/what-are-the-differences-between-the-threading-and-multiprocessing-modules> [Accessed 29 March 2021].
[8] Medium. 2021. Understanding Python Multithreading and Multiprocessing via Simulation. [online] Available at: <https://towardsdatascience.com/understanding-python-multithreading-and-multiprocessing-via-simulation-3f600dbbfe31> [Accessed 29 March 2021].
[9] PowerData, R., 2021. Spark vs Hadoop, ¿quién saldrá vencedor?. [online] Blog.powerdata.es. Available at: <https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/spark-vs-hadoop-quien-saldra-vencedor> [Accessed 22 May 2021].
[10] Youtube.com. 2021. EDB Team. [online] Available at: <https://www.youtube.com/watch?v=M26iIqmqWkI&ab_channel=EDteam> [Accessed 22 May 2021].
[11] Docs.python.org. 2021. Glossary — Python 3.9.5 documentation. [online] Available at: <https://docs.python.org/3/glossary.html#term-generator> [Accessed 22 May 2021].
[12] Npmjs.com. 2021. image processing - npm search. [online] Available at: <https://www.npmjs.com/search?q=image%20processing> [Accessed 22 May 2021].
[13] Docs.opencv.org. 2021. OpenCV: Modules. [online] Available at: <https://docs.opencv.org/master/modules.html> [Accessed 22 May 2021].
[14] GitHub. 2021. opencv/opencv_contrib. [online] Available at: <https://github.com/opencv/opencv_contrib/tree/master/modules/text> [Accessed 22 May 2021].
[15] [online] Available at: <https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_table_of_cntents_imgproc /py_table_of_contents_imgproc.html> [Accessed 23 May 2021].
[16] PyPI. 2021. pdf2image. [online] Available at: <https://pypi.org/project/pdf2image/> [Accessed 23 May 2021].
[17] Nodejs.org. 2021. Child process | Node.js v16.2.0 Documentation. [online] Available at: <https://nodejs.org/api/child_process.html> [Accessed 23 May 2021].