PROYECTO DE TITULACIÓN

SISTEMA PARA LA REVISIÓN DE EXÁMENES

PARA OBTENER EL TÍTULO DE

INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES.

PRESENTA:

TRISTAN NATHANIEL HUERTA VALDIVIA

ASESOR:

EDWIN CELESTINO GARCÍA ALCOCER

ÍNDICE

i.LISTA DE FIGURAS

ii.LISTA DE TABLAS

I.PRELIMINARES

I.1. Agradecimientos.

Agradezco a mi familia por el apoyo incondicional que me ofrecen día a día y sobre todo en los momentos no tan buenos y difíciles de la carrera como elaboración del proyecto, agradezco a mi mejor amigo por reducir mi estrés y darme consejo así como la felicidad que siempre necesito, agradezco a Ricardo Guadalupe Gómez Martínez por las explicaciones, pláticas interesantes, conocimientos, apoyo y momentos divertidos que hemos pasado, agradezco al Doctor Aldonso Becerra Sánchez por la oportunidad brindada, el conocimiento y ayuda en el proyecto, asimismo a mi asesor Edwin Celestino García Alcocer, por último agradezco a mis compañeras Estefanía Gallegos Silva, Jaqueline Garcia Luevano, Yatziri Amparo Esquivel Cruz y Mayguali Guadalupe Martínez Casillas por apoyarme, hacer más amena la carrera con ocurrencias graciosas, buenas explicaciones y muy buena amistad, gracias por todo, hasta por lo que no recuerdan, ¡Gracias!.

I.2. Resumen.

El reporte expuso el desarrollo de un sistema para la revisión de exámenes, por medio del uso del dataset (conjunto de datos) MNIST contiene más de 60,000 imágenes del sistema de numeración decimal, las imágenes se emplearon en el entrenamiento de una red neuronal artificial la que tuvo como objetivo lograr la detección de caracteres escritos a mano, específicamente dentro de una hoja de respuestas, generada con 50 campos rellenables, usando Python como lenguaje, Tensor Flow y Keras como librerías que facilitaron la elaboración de la red neuronal y un escáner de impresora para la digitalización de la hoja de respuestas llenada por los encuestados o alumnos.

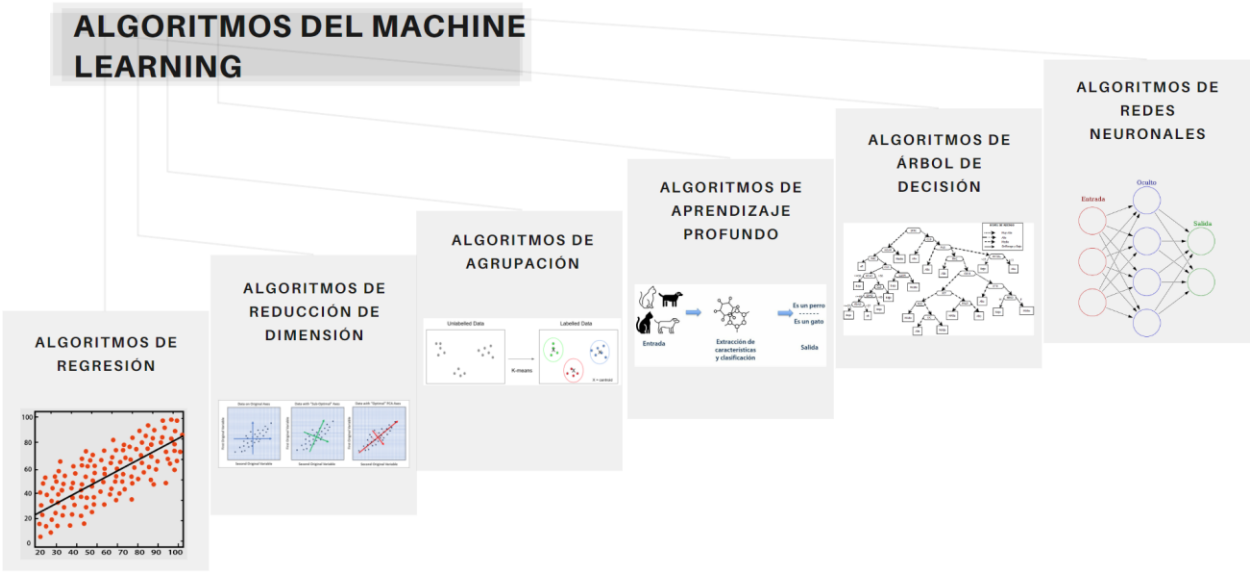
II.GENERALIDADES DEL PROYECTO

II.3. Introducción.

Todos los avances tecnológicos que intentan automatizar los procesos intelectuales que realiza el cerebro humano son llamados inteligencia artificial. El aprendizaje automático o machine learning es una técnica de inteligencia artificial, que imita la capacidad de aprender en una computadora empleando algoritmos tales como **Figura *1***:

Figura 1

Algoritmos de aprendizaje automatizado Fuente: Elaboración propia



Algunos de los algoritmos más reconocidos de este campo son el aprendizaje profundo o Deep Learning que es un subconjunto del aprendizaje automático por medio de redes neuronales con diferentes niveles de capas intermedias entre sus capas de entrada y salidas, por lo que se refiere a la profundidad de aprendizaje que es necesaria para un proceso inteligente como clasificación de imágenes, reconocimiento de imágenes, voz y escritura, reconocimiento muy similar al humano para predicción de movimientos, negocios, traducción de lenguas, conversaciones y responder en una conversación.

Generar un sistema aplicando algoritmos de redes neuronales para la revisión de exámenes, llega a ser importante para el avance en la educación, al automatizar un proceso como lo es la repetitiva acción de revisar exámenes provoca una mejora y a la vez una capacidad multitarea, dando paso a temas relevantes para la educación, como lo son la cobertura de la educación, la calidad, la gestión, los recursos e inversión de

infraestructura. El proceso a seguir para llegar al progreso deseado es por medio de sistemas que automaticen las acciones repetitivas, como el sistema para la revisión de exámenes quien aporta su avance tecnológico a la dirección de la mejora educativa.

II.4. Descripción de la empresa u organización y del puesto o área del trabajo del residente.

Las residencias profesionales fueron llevadas a cabo en el Laboratorio de Iluminación Artificial (LIA), se encuentra dentro del Tecnológico Nacional campus Pabellón de Arteaga, fundado el 01 de septiembre de 2008, ubicado en Carretera a la Estación de Rincón km 1, C.P. 20670 Pabellón de Arteaga, del estado de Aguascalientes.

Figura 2

Vista satelital del Instituto Tecnológico de Pabellón de Arteaga, Aguascalientes.

*Nota: Adaptado de Google Earth [Fotografía], por Maxar Technologies INEGI, 2019,(*Google Earth*). Todos los derechos reservados 2022 por Google.*

En 2017, el Dr. José Ernesto Olvera González y la Dra. Nivia Iracemi Escalante García fundaron el Laboratorio de Iluminación Artificial dentro del Tecnológico Nacional de México campus Pabellón de Arteaga gracias a apoyos recibidos por parte de la Convocatoria de Infraestructura Científica y Tecnológica del CONACyT (INFRA-2016-01, Project No. 270665). El equipo que se encuentra en LIA se ha logrado obtener gracias a

diversos apoyos federales y estatales (CB-2016-01, Project No. 287828, IDSCEA, SADER, por mencionar algunos.)

Figura 3

Diseño de LIA.

Las actividades realizadas en LIA permiten generar un impacto a nivel internacional debido a las herramientas tecnológicas. Existen cuatro áreas de investigación aplicada y desarrollo tecnológico enfocados a la línea de investigación de Biosistemas Mecatrónicos. A continuación, se presenta una breve descripción de las áreas de investigación. Dichos proyectos constan de las siguientes descripciones:

- Diseño, construcción e implementación de sistemas de producción multinivel en espacios cerrados con luz artificial LED que permiten potenciar el contenido nutrimental de cultivos como microgreens, lechuga, lenteja, espinaca, albahaca, alfalfa, entre otras, a través de recetas de luz (diferente s combinaciones de color).
- Desarrollo de Sistemas de desinfección de alimentos con radiación ultravioleta LED tipo A, B y C aplicados a productos agroindustriales y en fresco.
- Implementar estrategias tecnológicas con luz artificial tipo LED para preservar, extender y/o acelerar el tiempo de vida en almacén de frutas y verduras, además de analizar el efecto sobre la biosíntesis de compuestos (licopeno, capsaicina, entre otros.) durante su estancia en el anaquel.
- Investigación, integración y aplicación de la agricultura de precisión con el uso de tecnología aérea no tripulada (VANTs) para el monitoreo y detección de plagas en diferentes cultivos con el objetivo de evitar pérdidas en la producción.

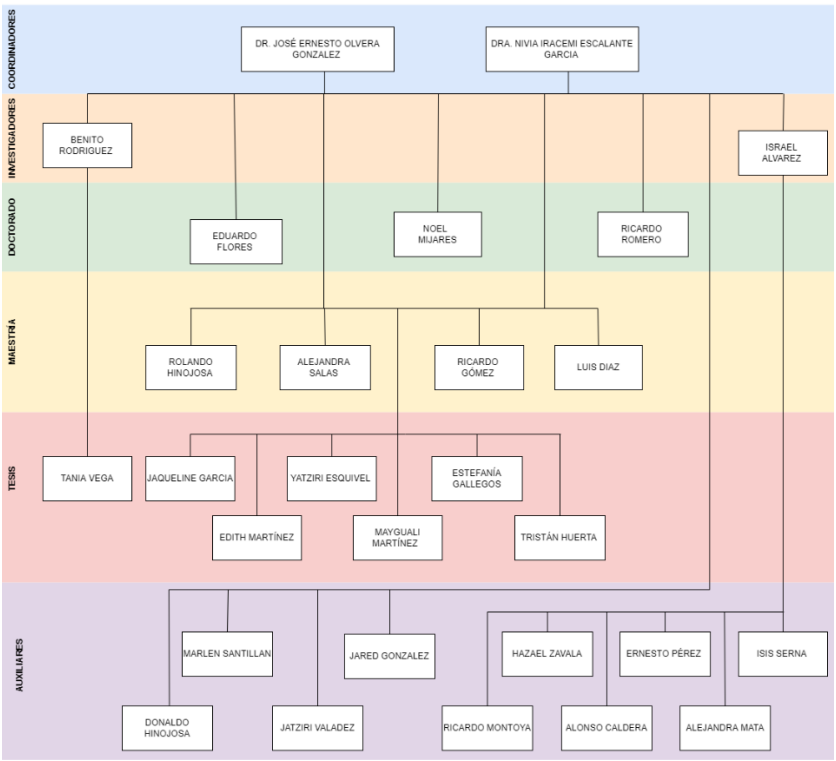
LIA cuenta con la siguiente infraestructura y es importante mencionar que LIA es el único Laboratorio en el País que se ha estado especializando en el uso de la luz artificial para el desarrollo de proyectos e investigaciones

En LIA se cuenta con 3 cámaras de crecimiento (descripción Temperatura 12C a 30C. Iluminación de área de trabajo. Humedad: Ambiente +5 % a 90 %). Programación día/noche de temperatura, racks y accesorios para producción de cultivos comerciales, sistema de caracterización de la luz (reflexión, absorción y transmisión) que incluye dos espectrofotómetros Ocean, esfera de medición de esparcimiento de luz e integradora (0.5 a 1 m) con accesorios para la manipulación de objetos en el interior. Un espectro radiómetro ILT950, flurómetro WALZ, SPAD 502, sensores de radiación fotosintéticamente activa, sensores climatológicos (humedad, temperatura, CO2 y radiación), Controladores Lógicos Programables, Sistema de radiación artificial tipo LED RGBW (rojo, azul, verde y blanco) de 0 a 1000 µmol de intensidad, con canal programable independiente para el color. Frecuencia de pulsado de 1 a 500 kHz. Con software y control independiente. Sistemas de radiación artificial equipados con V-LED e infrarrojo.

La estructura interna del laboratorio de iluminación artificial, así como los niveles jerárquicos en función del nivel de educación se hacen ver en la

Figura 4

Organigrama LIA.



MISIÓN

Proyectar e impulsar el desarrollo de nuevas actividades en el sector agroindustrial a través de procedimientos que ayuden a mejorar la calidad de vida alrededor del mundo, generando productos agroalimentarios con valor agregado.

VISIÓN

Ser un centro capaz de impulsar, desarrollar y fortalecer iniciativas que incentiven la productividad y rentabilidad agroalimentaria; generando en el sector agroindustrial nuevas filosofías de competitividad y calidad.

OBJETIVOS

Profundizar y clarecer los fenómenos no explicados hasta el momento sobre los efectos de la luz (continua y pulsada) en plantas y alimentos como su caracterización biológica con herramientas tecnológicas actuales LEDs, siendo un centro capaz de poseer el nivel de competitividad que otros laboratorios existentes en el mundo.

Debido a que el residente cursó la carrera de ingeniería en tecnologías de la información y comunicaciones el área de trabajo en la que se genera este proyecto en desarrollo tecnológico.

II.5. Problemas a resolver.

El Laboratorio de Iluminación Artificial (LIA) propuso la investigación para revisión de exámenes mediante la tecnología OCR (Reconocimiento Óptico de Caracteres), ya que, en un ámbito estudiantil, esta tarea manual de revisión conlleva mucho tiempo para los docentes. Provocando desperdicios de tiempo y esfuerzos en acciones repetitivas que una maquina o sistema logran de mejor manera. El averiguar si un sistema facilita la revisión de exámenes usando algoritmos machine learning es crucial para conocer el enfoque que el desarrollo tecnológico enfocado a la educación debería encausar.

Los problemas a resolver son calidad de revisión y tiempo de revisión, donde la correcta programación y enseñanza de un sistema que aprenda a revisar un examen causa mejores procesos de calidad, por el contrario un humano comete errores provocados por múltiples factores tales como sesgos perceptuales, cometidos comúnmente por problemas de visión; cansancio, distracciones, dejar cosas a medias e incluso generar cambios de respuestas o revisar de manera incorrecta, como castigo o venganza por

alguna agresión u ofensa de uno alumno, provocando una baja calidad de revisión, para evitar ello se empleará una red neuronal artificial, que será entrenada para reconocer números escritos a mano, obteniendo las respuestas contestadas por los alumnos para ser posteriormente revisadas con las respuestas correctas.

El tiempo de revisión mejora con la ayuda de la capacidad de procesamiento computacional actual, que supera sin problemas a un humano para tareas de comparación, reduciendo tiempos de revisión de un examen a lapsos sumamente menores, quitando del medio factores humanos como el cansancio visual, motriz y potencialmente la memorización. Es importante mejorar los tiempos de revisión para dar campo a tareas que mejoren el aprendizaje, por medio de multitasking dejando al sistema revisando exámenes y a su vez preparando la repartición de los próximos temarios (por dar ejemplo), el cómo se hará para reducir estos tiempos es digitalizando la revisión por medio de ciclos “FOR” y la comparación de las respuestas del alumno con las respuestas correctas.

Una hoja de respuestas común es capaz de albergar determinado número de cuadros de respuesta por las opciones múltiples que son ofrecidas en estas, en cambio un sistema que reconoce letras o números da la posibilidad a colocar una cantidad superior a la usual en una misma hoja de respuestas, el valor aquí es que es posible reducir el número de hojas de respuestas en exámenes largos, como lo son los exámenes nacionales de ingreso, el método será comenzar con pruebas de hojas de respuestas creadas especialmente para el sistema que contengan al menos cincuenta campos rellenables de números del sistemas de numeración decimal (del 0 al 9).

II.6. Justificación.

El área principal a quien se pretende abarcar fue a los maestros de distintos niveles educativos, debido a que estos son los principales usuarios puesto la cantidad de exámenes que estos llegan a revisar puede ser muy abundante.

El origen de la idea para la creación de un sistema que permite la revisión de exámenes, surge de la necesidad de agilizar este proceso, para dar espacio a encausar en un fin distinto como lo pueden ser la mejora del aprendizaje, enfocarse en aquellos temas en el que hace falta más tiempo. Tiempo que proporciona el sistema ahorrando la revisión

de exámenes, además por parte de los alumnos una espera de sus resultados mucho menor.

Es importante mencionar que este no solo se enfoca en un área específica de la educación, sino que este puede ser usado en cualquier tipo de evaluación para el que requiera de la revisión de una hoja de respuestas, como lo son escuelas, secundarias, preparatorias, e incluso áreas que no tengan que ver con la educación.

II.7. Objetivos.

OBJETIVO GENERAL:

Generar una herramienta tecnológica aplicando Redes Neuronales y deep learning con procesamiento de imágenes para la evaluación de exámenes con reconocimiento óptico de caracteres (OCR).

OBJETIVOS ESPECÍFICOS:

1. Determinar las desventajas que tiene llevar a cabo a la evaluación manual de revisión de exámenes.
2. Registrar los requerimientos generales para la integración de la aplicación.
3. Generar diagramas de diseño con diferentes arquitecturas de software para validar los procedimientos involucrados.
4. Disponer de lenguajes de programación y tecnologías para la generación del prototipo.
5. Evaluar y retroalimentar los errores que se presenten tanto en el BackEnd y FrontEnd.

III.MARCO TEÓRICO

III.8. Marco Teórico

Dentro de este apartado abordamos los conceptos usados para el diseño, desarrollo e implementación de la propuesta.

El reconocimiento óptico de caracteres (OCR) es una técnica que reconoce imágenes por medio de procesos que facilitan a la computadora o modelo neuronal a reconocer cualquier tipo de carácter como lo son números o abecedarios, por medio de otras técnicas como las que hacen saber en UAEM “binarización, segmentación, normalización, esqueletización y proyecciones” (Ángel et al., 2018).

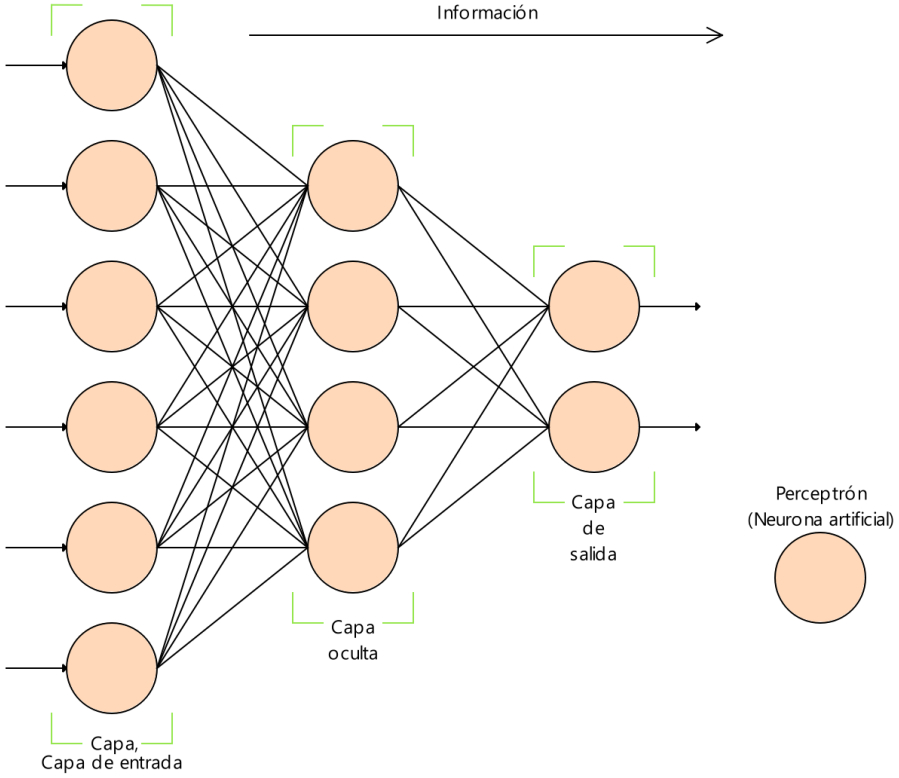
Para lograr que una máquina sea capaz de reconocer cualquier cosa necesita ser entrenada, en el caso del proyecto por medio de redes neuronales artificiales y datasets, además de librerías que ayudan a facilitar la creación de redes neuronales. Los datasets (conjunto de datos) son documentos de texto separados con comas y organizados como tablas, donde cada columna representa una variable en particular y cada fila a un miembro determinado del conjunto de datos que se tratan (DIMAS, 2021).

Existen diversos datasets fabricados con los datos limpios y de calidad, listos para ser usados para entrenar, como lo fue el dataset MNIST tiene 60,000 imágenes para entrenamiento y 10,000 para pruebas (Yann LeCun, 2022), de números del cero al nueve escritos a mano, por el que fue usado para el entrenamiento de la red neuronal del proyecto.

Las redes neuronales artificiales (RNA) buscan emular el funcionamiento del cerebro humano a través de unidades llamadas perceptrón (neurona artificial) y la conexión entre sí de un conjunto de estas, normalmente se organizan en capas como se ve en la **Figura 5**, (IBM, 2021)

Figura 5

Estructura de una red neuronal Fuente: Elaboración propia.



Como menciono el Grupo de Topología Computacional y Matemática Aplicada CATAM el 2022 en definiciones básicas de redes neuronales acerca de las capas donde coinciden en al menos tres:

- “Capa de entrada: compuesta por neuronas que reciben datos o señales procedentes del entorno.
- Capa de salida: aquella cuyas neuronas proporcionan la respuesta de la red neuronal.
- Capa oculta: aquella que no tiene una conexión directa con el entorno.”

Además, nos menciona que en la capa oculta se genera el procesamiento de la red neuronal, fue por esto que esta capa no se comunica con su entorno, similar a lo que hace un cerebro humano, que recibe estímulos del exterior (capa de entrada), los procesa (capas ocultas) y actúa conforme a ello (capa de salida).

Al entrenar una red neuronal con algunos ejemplos de entrenamiento e intentar minimizar el error lo más posible, se corre el riesgo de especializar demasiado nuestra red o mejor dicho generar un sobreajuste, la que generará que se comporte óptimamente con los datos entrenados, pero con ejemplos nuevos con los que no se ha entrenado puede llegar a tener errores serios (CATAM, 2022).

Durante el desarrollo de un sistema se hace uso de editores de texto los cuales facilitan la lectura e identificación de ciertos componentes, un ejemplo de editor fue el bloc de notas el más simple y básico, ya que este se encuentra por defecto al tener un sistema operativo Windows, existen otros que se enfocan en la edición de código tal como el usado en este proyecto “Visual Studio Code”, seleccionado por ser ligero y tener amplias funcionalidades así como extensiones que pueden ser instaladas sobre este para facilitar el entendimiento y lectura de ciertos códigos en diversos lenguajes de programación (Microsoft, 2021; Visual Studio Code, 2022).

Para el desarrollo de una red neuronal fue necesario un lenguaje de programación apropiado como lo fue Python, reconocido por lo efectivo y famoso que este fue en torno a inteligencia artificial, existen otros lenguajes de programación usados para creación de IA como los que se ven en la **Tabla 1¡Error! No se encuentra el origen de la referencia.**:

Tabla 1

Lenguajes de programación orientados a desarrollo de IA.

	Características					
Lenguaje	Simple	Conciso	Legible	Declarativo	Multiplataforma	Difícil curva de aprendizaje
Python	**X**	**X**	**X**	**X**	**X**	
C++			**X**		**X**	**X**
R		**X**		**X**	**X**	**X**
Java			**X**		**X**	**X**
Prolog				**X**		**X**

PYTHON

Un lenguaje de alto nivel interpretado, admite variedad de paradigmas de programación, como la estructurada, especialmente procedimental, orientada a objetos y funcional. Tiene como objetivo ser fácil de comprender, ser lógico para proyectos de diversos tamaños (Llerena Buenaño & Salazar Villamar, 2022).

Es posible generar una RNA por medio de Python sin ayuda de complementos o librerías, pero el hacer esto fue un error ya que se desperdiciaría el trabajo que otros ya han hecho, generando estructuras, funciones, programando lógicamente y aplicando fórmulas matemáticas. En la búsqueda de librerías que se enfoquen en la creación de redes neuronales se encontró Tensor Flow, una plataforma que ayuda a realizar modelos de aprendizaje automático sin la necesidad de ser un experto en ello, usado en páginas web, dispositivos móviles, producción y computadoras de escritorio (TensorFlow, 2022). Dentro de la enorme librería que fue Tensor Flow se tiene a Keras una interfaz de programación de aplicaciones o API que ayuda a reducir líneas de código simplificando acciones repetitivas dentro del proceso de creación de redes neuronales, haciendo que sea como menciona su sitio:” Sencillo. Flexible. Poderoso.” (Keras, 2022).

IV.DESARROLLO

IV.9. Procedimiento y descripción de las actividades realizadas.

Para el desarrollo del sistema optaron por el uso de una metodología ágil conocida como *Prototyping.* Esta metodología se define como un método de desarrollo de sistemas en el que se construye, prueba y luego se reinventa un prototipo (una primera aproximación de un sistema o producto final) cuanto sea necesario. El proceso continúa hasta que finalmente se obtiene un prototipo aceptable, a partir de este se puede desarrollar el sistema o producto completo. Prototipar hace posible obtener retroalimentación de algunas de las partes interesadas en el sistema en las primeras etapas, lo que da como resultado que las funcionalidades del sistema se descarten o supriman, agregando nuevas funcionalidades y requisitos a medida que se requieren. Prototipo, una versión preliminar, deliberadamente incompleta o reducida de un sistema (Alfredo et al., 2007).

El uso de prototipos fue una herramienta de utilidad que se puede aplicar a casi cualquier actividad de diseño y construcción de software. En las cuestiones tácticas de este trabajo, el prototipo diseñado se basará en las especificaciones de los requisitos bajo los supuestos de:

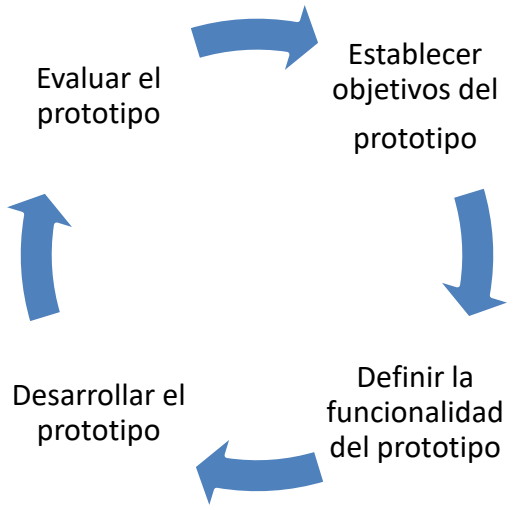
1. Definición de los objetivos del prototipo.
2. Validación de los requerimientos a través de la funcionalidad del producto.
3. Diseñar y desarrollar el prototipo.
4. Evaluar el funcionamiento del prototipo.

El enfoque de creación de prototipos sigue un proceso que se repite en cada iteración (De et al., 2018) (

Figura 6). Definir los objetivos del prototipo consiste en elegir una de las funcionalidades a desarrollar y determinar hasta dónde llegaría el prototipo en la fase de evaluación. Esta tarea requiere una cuidadosa selección de funciones que el usuario final debe evaluar para su aprobación o rechazo. Cada uno de estos pasos nos permite mejorar el sistema incluso antes de que esté terminado.

Figura 6

Ciclo de vida del prototipo.



La versión a la que el sistema llego se logró tras cruzar por el ciclo de prototipado cuatro veces, las cuales son descritas a continuación.

Trazado del ciclo I:

1. Definición de los objetivos del prototipo:
   Los objetivos establecidos fueron en función de generar la base la que soportaría todo el sistema, obteniendo:
   - Adaptación al entorno de trabajo: El entendimiento de los temas que involucran el desarrollo del sistema facilitan su implementación, es por ello que, se hicieron investigaciones de machine learning, redes neuronales artificiales, lenguaje de programación Python y librerías.
   - Generar experiencia por medio del uso de sistemas relacionados: Al cumplir con el objetivo se adquirió un conocimiento más empírico de lo que fue capaz un sistema que hace uso de tecnologías como lo son las RNA y machine learning.
2. Validación de los requerimientos a través de la funcionalidad del producto:
   Los requerimientos dados a validad a las autoridades correspondientes fueron:
   - Generar pruebas de sistemas OCR funcionales, asignada para cumplirse en dos quincenas

- Proveer al sistema probado imágenes de entrada generadas personalmente, asignada para cumplirse en una quincena

3. Diseñar y desarrollar el prototipo:

   Para el cumplimiento de los objetivos como los requerimientos dados, se generó investigación de sistemas que hicieran uso de OCR, de ser posible sistemas que fueran capaces de ser descargados y probados con datos proporcionados por agentes externos al sistema.

4. Evaluar el funcionamiento del prototipo:

   El prototipo inicial fue una red neuronal artificial, quien fue capaz de recibir datos de entrada de cualquier tipo, donde era procesados por la red para que esta dijese el número suministrado, dando una evaluación positiva por parte de las autoridades correspondientes.

Trazado del ciclo II:

1. Definición de los objetivos del prototipo:

   Los objetivos establecidos fueron en función de generar adaptación del prototipo previamente obtenido y los objetivos de revisión como uso de una hoja de respuestas en el sistema:

   - Generar una hoja de respuestas para el sistema: Este objetivo pretende facilitar el entendimiento del sistema, así como el hacer más propio el sistema.
   - Facilitar el recorte de la hoja de respuestas: El objetivo pretende lograr una extracción correcta de las respuestas dadas en la hoja de respuestas, ya que fue crucial para el correcto funcionamiento de la red neuronal artificial.

2. Validación de los requerimientos a través de la funcionalidad del producto:

   Los requerimientos dados a validad a las autoridades correspondientes fueron:

   - Generar una hoja de respuestas intuitiva y estandarizar el recorte de la hoja de respuestas, asignada para cumplirse en dos quincenas.

3. Diseñar y desarrollar el prototipo:

   Para el cumplimiento de los objetivos como los requerimientos dados, se realizó una pequeña comparación de diversas hojas de respuestas de exámenes, para el

que ayudo a reconocer las características comunes que estas tienen, para así generar una hoja de respuestas propia del sistema. Además, se hizo uso de la librería PILLOW o PIL (librería de código abierto para Python, quien facilita el abrir, manipular y guardar muchos formatos de archivos de imágenes distintas) para generar el recorte y automatización del mismo.

4. Evaluar el funcionamiento del prototipo:

   Se obtuvo la hoja de respuestas adecuada al sistema y un sistema de automatización de los recuadros de respuesta quienes fueron evaluados positivamente por las autoridades correspondientes

Trazado del ciclo III:

1. Definición de los objetivos del prototipo:

   Los objetivos establecidos fueron en función de generar las predicciones con la RNA de todos los recuadros de respuesta obtenidos del prototipo anterior, obteniendo:

   - Preparar el entorno de desarrollo en Visual Studio y haciendo uso del lenguaje de programación Python: El objetivo pretende facilitar la ejecución de la RNA, adaptándola a un entorno de fácil ejecución.
   - Generar la arquitectura del proyecto: La planeación del funcionamiento del sistema ayuda al proceso de desarrollo, siendo esto lo que el objetivo pretende abarcar.

2. Validación de los requerimientos a través de la funcionalidad del producto:

   Los requerimientos dados a validad a las autoridades correspondientes fueron:

   - Generar entorno para RNA con Python, asignada para cumplirse en dos quincenas junto a la planeación de la arquitectura del sistema para la revisión de exámenes.

3. Diseñar y desarrollar el prototipo:

   Para el cumplimiento de los objetivos como los requerimientos dados, se generó la instalación de los programas y librerías necesarias para que la RNA fuese capaz de ejecutarse y funcionar correctamente en una computadora de uso personal y no un servidor local como anteriormente se hizo uso de esta.

4. Evaluar el funcionamiento del prototipo:

   Se logró obtener un entorno adecuado para el código de la red neuronal artificial y este fue aprobado por las autoridades correspondientes.

Trazado del ciclo IV:

1. Definición de los objetivos del prototipo:

   Los objetivos establecidos fueron en función de facilitar el uso de la RNA obtenida y adaptada para usarse en Visual Studio Code, así como el generar el funcionamiento que le da nombre al sistema y la revisión, obteniendo:

   - Conocer como se hace reúso de una RNA: Este objetivo pretende aminorar el tiempo de ejecución de la RNA por medio del reúso de la RNA ya entrenada.
   - Desarrollar la revisión del examen con el uso de las respuestas correctas dadas y las respuestas procesadas por la RNA: este objetivo pretende terminar el con un prototipo del sistema completo, por el que logre revisar un examen por medio del sistema para la revisión de exámenes.

2. Validación de los requerimientos a través de la funcionalidad del producto:

   Los requerimientos dados a validad a las autoridades correspondientes fueron:

   - Generar el guardado de una RNA entrenada.
   - Generar la revisión del examen tras el análisis por la RNA.

3. Diseñar y desarrollar el prototipo:

   Para el cumplimiento de los objetivos como los requerimientos dados, se generó una investigación para el que se hace uso de una red neuronal en sistemas cotidianos, ayudando a emplearla en el sistema para la revisión de exámenes, además se generaron pruebas para el uso de listas y diccionarios en Python para generar el código necesario para la revisión del examen.

4. Evaluar el funcionamiento del prototipo:

   El prototipo final obtenido fue un éxito, esto debido a que fue capaz de generar la revisión de la hoja de respuestas de un examen de manera exitosa, llegando a obtener la primera versión funcional del sistema para la revisión de exámenes, evaluado de manera positiva por las autoridades correspondientes.

V.RESULTADOS

V.10. Resultados

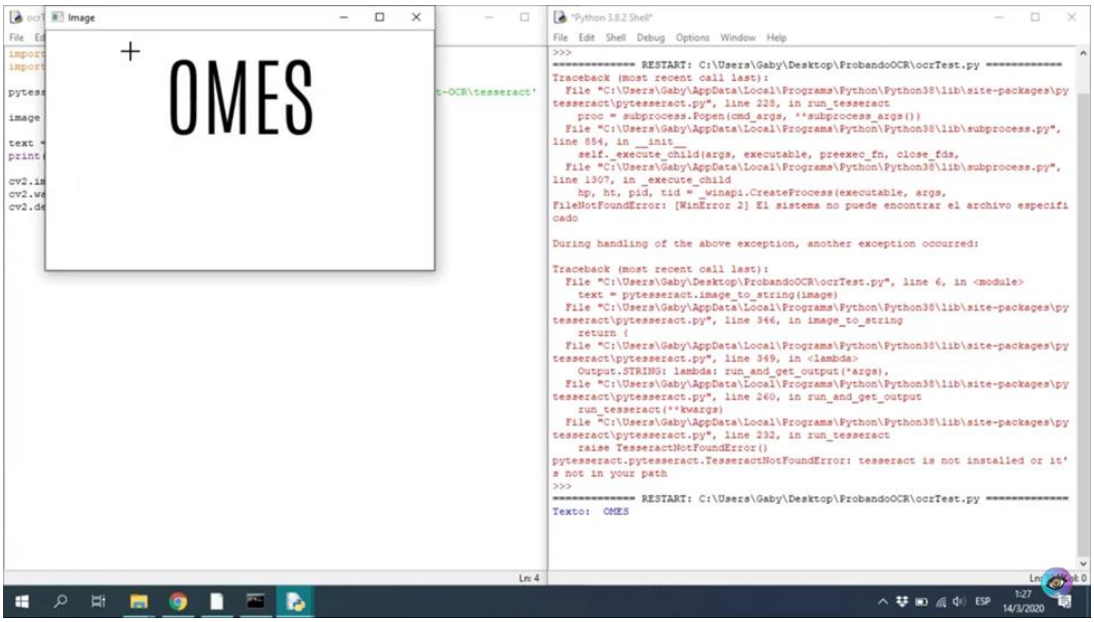
Resultados del trazado del ciclo I:

Se hizo cumplimiento de los objetivos propuestos por medio de la búsqueda de los fundamentos de machine learning, OCR, RNA, además, se generaron pruebas de sistemas con tecnología OCR.

Existen diversos sistemas los cuales hacen uso de OCR, sin embargo, uno de los problemas con los sistemas encontrados fue el tipo de reconocimiento que hacen, estos son capaces de reconocer palabras completas como se ve en la **Figura 7**

Figura 7

Sistema con OCR - Tesseract y Pytesseract.



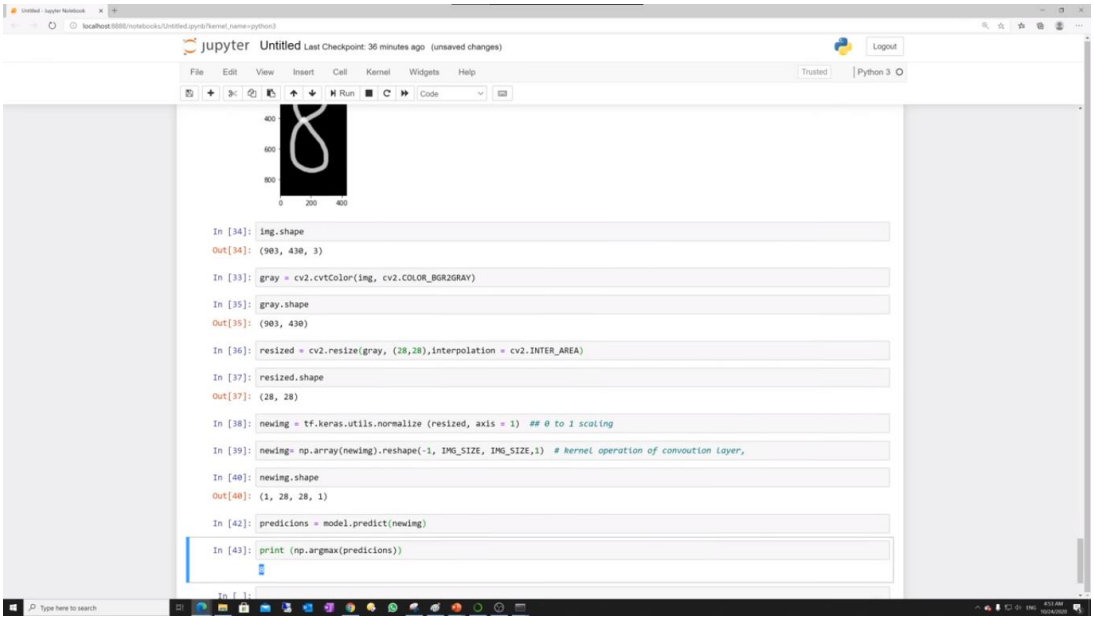
Nota: OMES, 2020. Como instalar Tesseract y Pytesseract en Windows | OpenCV OCR en Python [Archivo de Vídeo]. YouTube. https://www.youtube.com/watch?v=_j6o2rBkhhg&ab_channel=OMES

El problema encontrado fue el no logran reconocer letras por si solas, Tesseract fue capaz de reconocer palabras, en la **Figura 7** se pude ver como se le dio una imagen de entrada que contiene la palabra “OMES” el que fue el nombre del canal que hace la prueba de Tesseract. Al momento de hacer uso de Tesseract, este no fue capaz de

reconocer letras individuales, ni números. Esto provoco cambios en la búsqueda de sistemas OCR, llegando a las redes neuronales artificiales (RNA), descubriendo un video demostrativo del uso y creación de una RNA entrenada para detectar números escritos a mano; en el caso del video, con números creados con Paint.

Figura 8

Handwritten Digits Recognition Sistem.



Tras haber buscado y probado diversos sistemas de RNA y OCR, los cuales requieren un entorno distinto, así como específico para ser probados, se encontró que como base para el comienzo del sistema para la revisión de exámenes. En la **Figura 8** se logra observar el sistema (sistema para el que fue entrenado con el dataset MNIST contiene más de 60,000 imágenes del sistema de numeración decimal escrito a mano, con imágenes limpias y listas para ser usadas) fue capaz de recibir una imagen de un numero generado en Paint con el mouse de un computador, para después hacer uso de la red neuronal artificial previamente entrenada, dando como resultado de la red el decir que fue un “8”.

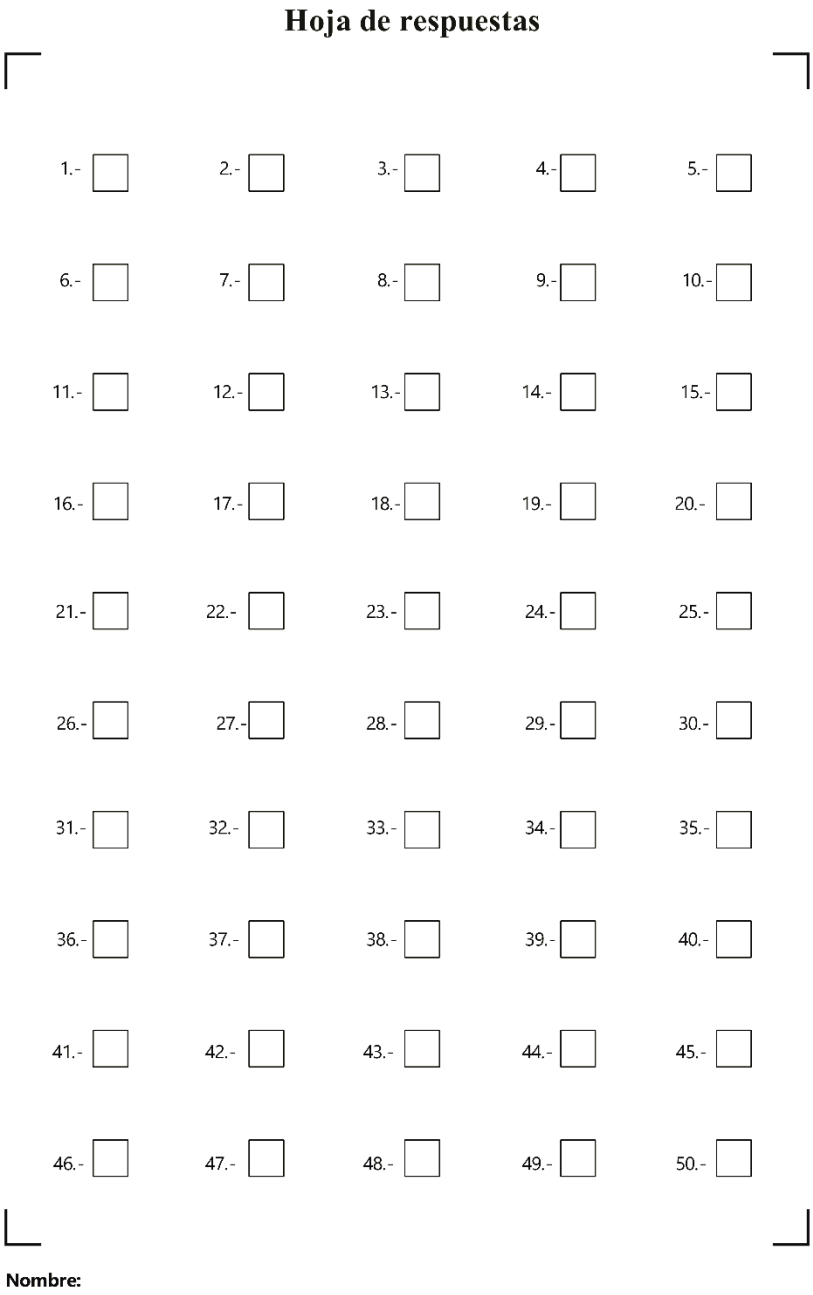
Resultados del trazado del ciclo II:

Se obtuvo la hoja de respuestas (

Figura 9) con 50 campos rellenables

Figura 9

Hoja de respuestas del sistema para la revisión de exámenes Fuente: Elaboración propia



El objetivo durante la creación de la hoja de respuestas fue el ser entendible para el usuario encuestado y minimalista, además de esto que los recuadros rellenables fueran simétricos entre sí, tanto vertical como horizontalmente, debido a que esto favoreció la automatización del recorte de los recuadros de respuesta.

Figura 10

Código prueba del recorte automatizado

sistema.py > ...

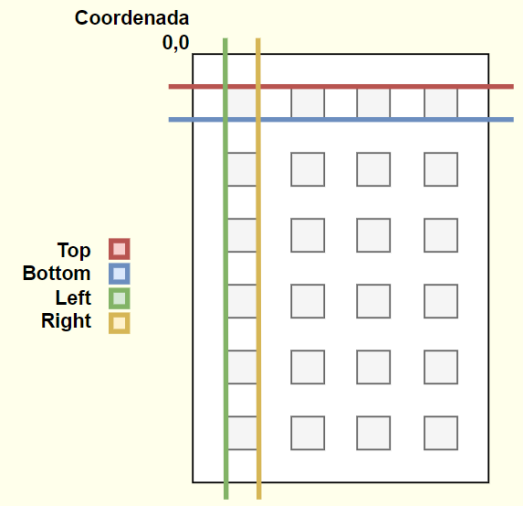
```python
from PIL import Image, ImageChops #Importar partes de librería PIL

im = Image.open(r"HRLLN-Prueba0.jpg") #Abriendo imagen escaneada de hoja de respuestas
im = im.resize((2481,3506)) #Redimencionamiento de imagen para evitar errores
imv = ImageChops.invert(im) #Invercion de colores de la imagen

t, b, l, r, x, x2 = (475, 550, 567, 2138, 0, 0) #Coordenadas iniciales y X para nombre de img. de cortes horizontales
l2, t2, r2, b2, y, reboot = (0, 0, 75, 75, 100, 0) #Coordenadas iniciales y X2 para nombre de img. de cortes verticales

##### Corte Horizontal #####
for i in range(10): #Para i en un rango de 10 has:
    im1 = imv.crop((l, t, r, b)) #Corte de imagen invertida y guardada en variable im1
    #im1.show() #Mostrar im1 en pantalla
    t = t + 262 #Suma de coordenadas para corte superior
    b = b + 262 #Suma de coordenadas para corte inferior
    x = x +1 #Suma de valor x para asignación de nombre .jpeg
    im1.save(str(x)+".jpeg") #Guardado de imagen cambiando nombre con variable x

    ##### Corte Vertical #####
for s in range(10): #Ciclo para pasar por las 10 imágenes recortadas de rangos de 5 recuadros (imagen 1x5)
    x2 = x2 + 1 #Variable que cambia el nombre de la imagen
    for u in range(5): #Ciclo para pasar por cada recuadro de la imagen 1x5
        if reboot == 5: #Cuando la variable llega a 5 se reinician los puntos de recorte
            l2, t2, r2, b2 = (0, 0, 75, 75) #Puntos de recorte
            reboot = 0 #Variable de reinicio
        imh = Image.open(str(x2)+".jpeg") #Se abre la imagen de los cortes horizontales
        im2 = imh.crop((l2, t2, r2, b2))  #Se genera el corte y se guarda en la variable im2
        #im2.show() #Se muestra el corte
        l2 = l2 + 375 #Suma de píxeles para ir cambiando de recuadro en recuadro
        r2 = r2 + 375 #Suma de píxeles para ir cambiando de recuadro en recuadro
        y = y + 1 # Se genera la suma para cambiar el nombre de la imagen guardada, imagen de recuadro recortado
        reboot = reboot + 1 #Se suma una vuelta al ciclo para el reinicio y posterior cambio de imagen 1x5
        im2.save("RL/"+str(y)+".jpeg") # Se guarda el recorte de la imagen de un recuadro de un solo número (imagen 1x1)
```

En la **Figura 10** se puede ver el código final (antes de ser acoplado al sistema entero) que genera el recorte automatizado de los 50 recuadros de respuesta de la hoja de respuestas (

Figura 9). En este se puede ver todo el funcionamiento del sistema, por el que fue descrito y se distingue por su color verde en donde abarca una gran área del código.

Figura 11

Recorte de un solo recuadro con PIL Fuente: Elaboración propia

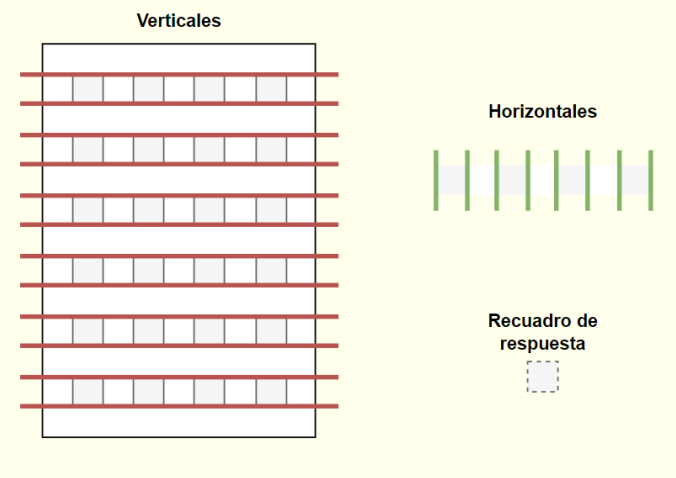


En la

Figura 11 se logra entender la librería PIL hizo los recortes de manera individual, esta comienza en la coordenada (0,0) la cual se encuentra en la parte superior izquierda y a partir de esta se le dan cuatro campos o coordenadas, correspondientes al alto del recorte (Top), la parte inferior del recorte (Bottom), la parte izquierda y derecha del mismo (Left & Right).

Figura 12

Funcionamiento del sistema de recorte automatizado Fuente: Elaboración propia



Para facilitar el colocar manualmente las 4 coordenadas 50 veces se hizo uso de ciclos “FOR” y una hoja de respuestas donde fue llenada simulando un caso real de llenado para su posterior uso en pruebas en la RNA.

El sistema de recorte automatizado se hace ver en la **Figura 12**, dicho sistema recibe la hoja de respuestas llena y previamente escaneada por un escáner, invierte los colores de la imagen facilitando el futuro procesamiento a la RNA (debido a que procesa los números solo en color blanco y fondo negro), luego genera un recorte de la primera fila de las respuestas y guarda el recorte generado, para posteriormente continuar recortando la misma imagen original de la hoja de respuestas completa pero en las coordenadas de la siguiente fila, continuar así hasta llegar a la última fila terminando con todos los cortes verticales guardados en imágenes llamadas según su número de fila.

Figura 13

Recorte vertical.

En la

Figura 13 podemos ver el resultado real de uno de los recortes verticales que genera el sistema de recorte automatizado. Luego de los recortes verticales el sistema vuelve a empezar los recortes, pero ahora con las tiras resultantes y generando recortes verticales como en la

Figura 9 terminando con los recuadros de respuesta listos para pasar por la RNA ya entenada, por el que son guardados con el nombre del número de respuesta que son más 100 como se ve en la **Figura 14** correspondiente al recuadro de respuesta número 1.

Figura 14

Recuadro de respuesta "101.jpeg"

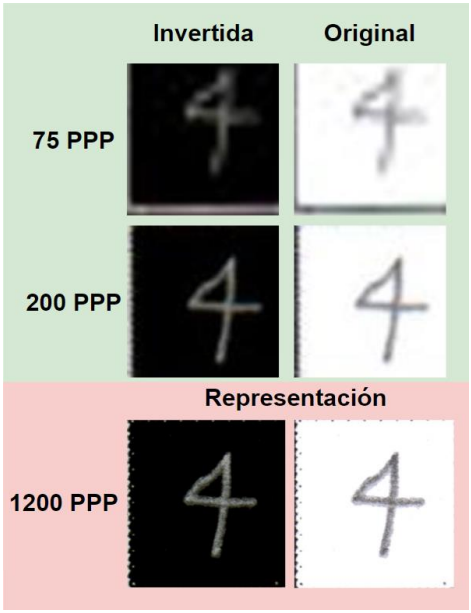
Luego de finalizar con el sistema de recorte automatizado, se generaron diversas pruebas con distintas calidades de escaneo, esto para conocer las limitaciones y posibles requerimientos que el sistema necesita para su funcionamiento, cabe mencionar que la **Figura *1*Figura 13** y la **Figura 14** se originan de la misma hoja de respuestas con una resolución de 200 pixeles por pulgada (PPP).

En estas pruebas se escaneo una misma hoja de respuestas con tres calidades distintas **Figura 15**, comenzando por un extremo mínimo de 75 PPP el cual se puede ver un tanto distorsionado lo que podría significar un problema de procesamiento por parte de la RNA, además de que el redimensionar su tamaño al estandarizado para el sistema, genera una falla en el recorte dejando a la vista en la parte inferior una línea del recuadro en el que se colocó el número, después se continuo con la calidad mas usual de escaneo 200 PPP la cual es la mejor opción debido a que proporciona la resolución necesaria, con ello el sistema de recorte automatizado fue calibrado y no causa algún tipo de fallo de procesamiento en la RNA, por último la resolución más alta fue de 1200 PPP la cual fue un problema desde el redimensionamiento, sobrepasando los pixeles admitidos de la librería PIL, la cual confundió con un ataque Bomb DOS mostrando el siguiente mensaje traducido a: El tamaño de la imagen (139203948 píxeles) excede el límite de 89478485 píxeles, podría ser un ataque de bomba de descompresión DOS. advertencias (mensaje original "*Image size (139203948 pixels) exceeds limit of 89478485 pixels, could be decompression bomb DOS attack.warnings"*) para continuar con la representación de la

Figura 15 se generó la captura de pantalla, que es mostrada en la parte inferior, por tanto se recomienda el uso de una calidad de 200 pixeles por pulgada.

Figura 15

Recortes re recuadros de respuesta a diferentes PPP Fuente: Elaboración propia



Resultados del trazado del ciclo III:

Tras realizar los requerimientos dados, se logró obtener la arquitectura del sistema para la revisión de exámenes donde se puede observar cómo se pretende que el sistema final funcione.

Figura 16

Arquitectura del sistema para la revisión de exámenes Fuente: Elaboración propia

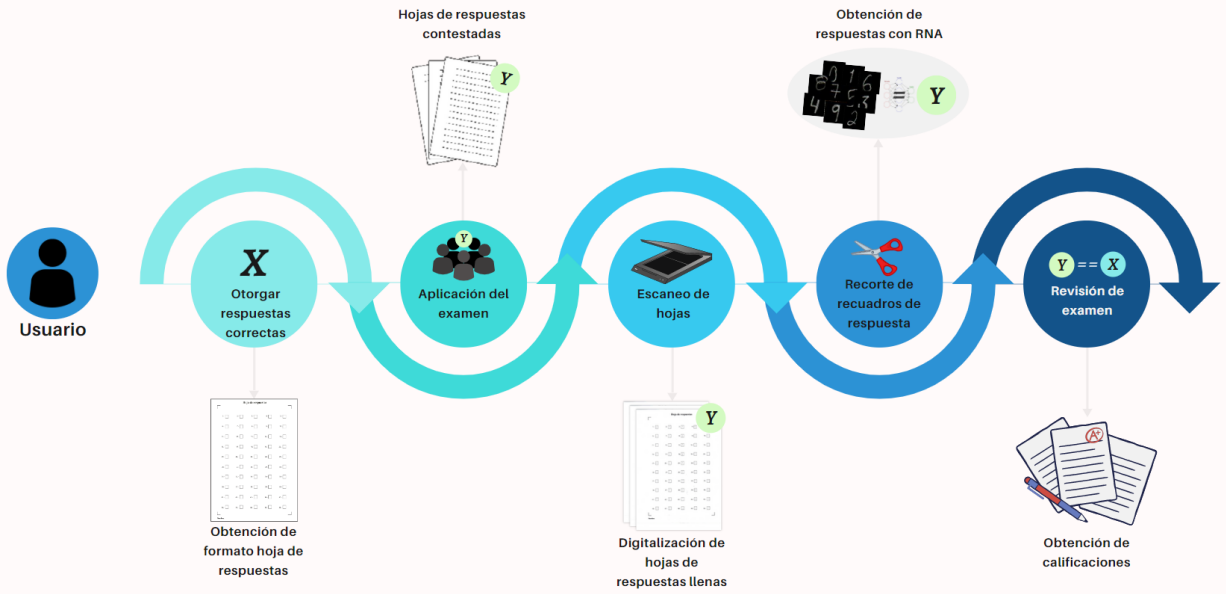


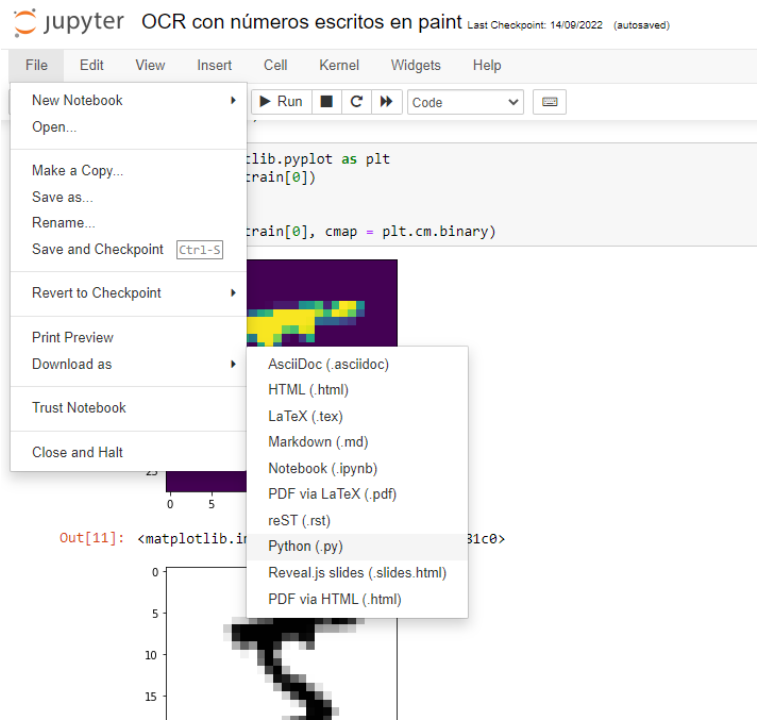
Figura 16, el usuario ejecuta el sistema y le dice el número de preguntas que su examen tendrá, para posteriormente dar al sistema todas las respuestas (únicamente con número del 0 al 9), el sistema le guardar a la hoja de respuestas en su computador para que sea impreso y aplicado a sus alumnos, después de esto las hojas de respuesta contestadas deberán de ser escaneadas por un escáner y ser entregadas al sistema, a fin de generar el recorte de los recuadros de respuesta y pasarlos por la RNA para generar la calificación del examen, por medio del conteo de las respuestas correctas e incorrectas que son mostradas en la interfaz del sistema.

La preparación del entorno se generó por medio de la instalación de Visual Studio, y la herramienta Pipenv que ayuda a generar entornos para aislarlos de otros y no generar problemas entre estos, donde se hizo instalación de todas las librerías que hacen que la RNA y el sistema de recorte automatizado fuesen funcionales, como:

- *Matplotlib: Facilita los procesos matemáticos, así como interpretaciones de los mismos*
- *Opencv-python: Ayuda a facilitar el manejo de las imágenes durante el proceso de predicción de la red neuronal artificial*
- *TensorFlow: Facilita la creación de todo tipo de redes neuronales artificiales*

Figura 17

Obtención de código fuente a formato de archivo ".py" Fuente: Elaboración propia.



Posterior a esta creación del entorno se descargó el código anterior (

Figura 17) que se contenía en Jupyter Notebook en un formato de archivo ".py" este facilito el proceso de adaptación y acople de todo el sistema.

Resultados del trazado del ciclo IV:

Los resultados obtenidos al finalizar el trazado de este ciclo fueron el lograr guardar la RNA ya entrenada, esto acorto los tiempos de ejecución de la misma a menos de un minuto, mejorando todo el proceso de predicción de la red, debido a que anterior a esta mejora, el sistema tardaba más de cinco minutos en generar el entrenamiento de la RNA para solo ser usada una vez para una predicción.

A esto se le llama guardado y serialización de modelos por medio de TensorFlow Keras, como mencionan en la página de TensorFlow en su sección de aprender, "*Puede guardar un modelo creado con la API funcional en un solo archivo. Posteriormente, puede volver a crear el mismo modelo a partir de este archivo, incluso si ya no tiene acceso al código que creo el modelo.*" Justo lo que necesario para la mejora e implementación correcta del sistema, pasando de tener 232 líneas de código solo para el entrenamiento y un solo uso de la RNA, a tan solo 30 líneas de código.

Figura 18

Código de RNA entrenada y escala del mismo en comparación del código anterior Fuente: Elaboración propia

PruebaRNAentrenada.py > ...

```
from __future__ import absolute_import, division, print_function, unicode_literals
from borrable import * # Traemos bariables

import cv2
import numpy as np
import tensorflow as tf
from tensorflow  import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPooling2D

NRR = 0
RERNAO = []

tf.keras.backend.clear_session()  # Para restablecer fácilmente el estado del portátil.

model = keras.models.load_model('modelo_entrenado.h5')
for i2 in range(NDP):
    NRR = NRR + 1
    img = cv2.imread(str(NRR)+".jpeg")
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    resized = cv2.resize(gray, (28,28),interpolation = cv2.INTER_AREA)
    resized.shape
    newimg = tf.keras.utils.normalize (resized, axis = 1)
    IMG_SIZE=28
    newimg = np.array(newimg).reshape(-1, IMG_SIZE, IMG_SIZE,1)
    newimg.shape
    predictions = model.predict(newimg)
    print (np.argmax(predictions))
    RERNAO.append(predictions)
    print(RERNAO)
```

En la

Figura 18 podemos observar en la parte derecha resaltado con un contorno verde tenemos la escala del código actual también se puede ver claramente en la parte izquierda de la figura y resaltado con un contorno rojizo, tenemos la escala del código anterior el que es casi siete veces más grande que el actual acoplado al sistema completo.

Para la revisión del examen se hizo uso de listas, por el que guardan las respuestas de los encuestados, respuestas procesadas por la RNA, también se hizo uso de diccionarios quienes guardan las respuestas correctas del examen, así como el numero de la pregunta para el que corresponden.

Figura 19

Sistema que genera la revisión de respuestas correctas con respuestas dadas por el usuario tras pasar por la RNA

RDE.py > ...

```python
from multiprocessing.sharedctypes import Value
########## --------------- RERNA sera remplasado por ls datos que proporcione la RNA --------------- ##########
RERNA = [1,2,3,4,5,6,7,8,9,0]# Respuesta encuestado RNA
NDP = int(input("¿Cuántas preguntas tendrá? ")) # NDP = Número de preguntas

PCRC ={} # PCRC = Preguntas con respuestas correctas
for n in range(1,NDP+1):
    PCRC[n] = int(input("Respuesta a pregunta número {}: ".format(n)))

print(PCRC)

for i in range(1,NDP+1):

    if RERNA[i-1] == PCRC.get(i):
        print('Respuesta {} correcta'.format(i))
    else:
        print('Respuesta {} incorrecta'.format(i))
```

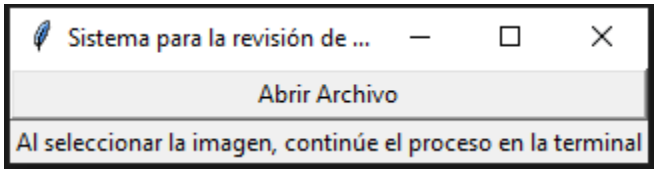
El sistema se probó obteniendo (ver

Figura 19), este es el código base antes de adaptarse a las demás partes del sistema, como la RNA y el sistema de recorte automatizado, en este se puede ver el sistema le pregunta al usuario ¿Cuantas preguntas necesita?, esto para guardarlo en una variable y posteriormente usarlo en un ciclo, este se repetirá según el número de preguntas que dio y preguntara ¿Cuál es la respuesta de “X” pregunta? Para posteriormente pasar a comparar una a una diciendo si son iguales con una impresión en la terminal diciendo “Respuesta X correcta” o “Respuesta X incorrecta” según sea el caso.

Por último, se generaron dos cosas, la pequeña interfaz por el que genera la comunicación con el usuario y es por este medio donde el usuario le da al sistema la hoja de respuestas llena y escaneada.

Figura 20

Interfaz del sistema para la revisión de exámenes Fuente: Elaboración propia.



En la **Figura 20** se puede ver la pequeña interfaz que pretende ser mejorada, pero para un funcionamiento inicial de prototipo es más que suficiente, esta interfaz únicamente contiene un botón, en donde permite al usuario seleccionar específicamente la hoja de repuestas que va a pasar por el sistema y también contiene un texto que le indica a donde dirigirse para continuar con el proceso del sistema para la revisión de exámenes.

Figura 21

Código a escala del sistema para la revisión de exámenes

Por último, el juntar las partes del sistema, iniciando con la importación de las librerías que todas las partes requieren, continuando con la ventana de la interfaz y las variables que se usaran para los ciclos “FOR”, luego el sistema de recorte automatizado que genera el corte de los 50 recuadros de respuesta de la imagen que el usuario seleccionó. Mientras tanto se continua con una parte del código de revisión, quien pregunta ¿Cuantas preguntas tendrá el examen? y pide las respuestas de cada una de estas, para continuar con la RNA reutilizable que procesa solo el número de recuadros de respuesta que van a ser usados (esto se sabe por el número de preguntas que el examen tendrá) y guarda cada predicción en la variable de respuestas del encuestado o alumno para posteriormente continuar con la revisión, donde se da el conteo de respuestas correctas e incorrectas y la calificación de la hoja de respuestas, todo esto en 120 líneas de código representadas a escala en la **Figura 21**.

VI.CONCLUSIONES

VI.11. Conclusiones del Proyecto

En este informe técnico se ha dado el primer paso a una solución basada en deep learning y redes neuronales para la evaluación de exámenes con el uso de reconocimiento óptico de caracteres, el sistema continua en desarrollo pero tiene el potencial de sustituir la forma en la que los exámenes son revisados actualmente por cualquier profesor, debido a que este sistema es de uso libre y pretende ser usado por cualquier persona y no solo por instituciones con los recursos suficientes para la obtención de una maquina que revisa exámenes de opción múltiple comunes.

En virtud de lo argumentado se puede conocer que el generar manualmente la revisión de exámenes es algo que cambiara, por los fallos humanos que cometemos y limitaciones que tenemos, dado que los requerimientos computacionales solicitados para una aplicación que automatiza esta acción son bajos y alcanzables para cualquiera con acceso a internet, dándole el acceso a estos lenguajes y tecnologías que permiten dar entrada a estas arquitecturas de software que pueden ser validadas y retroalimentadas por expertos en estas ciencias.

VII.COMPETENCIAS DESARROLLADAS

VII.12. Competencias desarrolladas y/o aplicadas.

1. Utilicé tecnologías y herramientas actuales y emergentes acordes a las necesidades del entorno.
2. Integré las diferentes arquitecturas de hardware y administré plataformas de software para incrementar la productividad en la organización.
3. Observé los aspectos legales del uso y explotación de las Tecnologías de la Información y Comunicaciones.
4. Integré soluciones basadas en sistemas de comunicaciones que involucren tecnologías actuales y emergentes.
5. Gestioné eficientemente los recursos de la organización con visión compartida, con el fin de suministrar bienes y servicios de calidad.
6. Desempeñé funciones de consultoría y auditoría en el campo de las Tecnologías de la Información y Comunicaciones.
7. Apliqué métodos de investigación para desarrollar e innovar modelos, sistemas, procesos y productos en las diferentes dimensiones de la organización.

VIII.FUENTES DE INFORMACIÓN

VII.13. Fuentes de información

Alfredo, P., Ridel, W., & Buemo, S. G. (2007). *Capítulo 12 Ingeniería de software: el proceso para el desarrollo de software*.

Ángel, G. M., Segura, Y. Y. C., Burlak, & G. (2018). *Reconocimiento de caracteres mediante OCR (Optical Character Recognition)*. http://www.progmat.uaem.mx:8080/Vol10num1/vol10num1art6.pdf

CATAM. (2022). *CONCEPTOS BÁSICOS SOBRE REDES NEURONALES*. http://grupo.us.es/gtocoma/pid/pid10/RedesNeuronales.htm

De, S., De, A., De, C., Becerra Sánchez, A., Valles, G. Z., Autónoma De Zacatecas, U., Ramírez, U., Correa, G., & Guerrero, S. E. (2018). LEARNING CONTENT MANAGEMENT SOFTWARE PERSONALIZED FOR A UNIVERSITY ENVIRONMENT (SOFTWARE DE ADMINISTRACIÓN DE CONTENIDOS DE APRENDIZAJE PERSONALIZADO PARA UN AMBIENTE UNIVERSITARIO). *Pistas Educativas*, *40*(130), 2448. http://pistaseducativas.celaya.tecnm.mx/index.php/pistas/article/view/1733

DIMAS. (2021). *¿QUÉ es un DATASET en PYTHON? - Curso Python PANDAS desde 0 - YouTube*. https://www.youtube.com/watch?v=CFPQGpzJJdQ&ab_channel=Dimas

IBM. (2021). *Conceptos básicos de nodos de modelado - Documentación de IBM*. https://www.ibm.com/docs/es/spss-modeler/saas?topic=overview-modeling-nodes

Keras. (2022). *Keras: the Python deep learning API*. https://keras.io/

Llerena Buenaño, A. S., & Salazar Villamar, M. J. (2022). *Automatización de un sistema identificador y posicionador de objetos a través de un Brazo robótico mediante Visión Artificial con Lenguaje Python*. http://dspace.ups.edu.ec/handle/123456789/22832

Microsoft. (2021). *Python en Windows para principiantes | Microsoft Learn*. https://learn.microsoft.com/es-es/windows/python/beginners

OMES. (2020). *Como instalar Tesseract y Pytesseract en Windows | OpenCV OCR en Python - YouTube*. https://www.youtube.com/watch?v=_j6o2rBkhhg&ab_channel=OMES

TensorFlow. (2022). *Introducción a TensorFlow*. https://www.tensorflow.org/learn?hl=es-419

Visual Studio Code. (2022). *Documentation for Visual Studio Code*. https://code.visualstudio.com/docs

Yann LeCun. (2022). *MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges*. http://yann.lecun.com/exdb/mnist/

IX.ANEXOS