UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

título del TFG
Documentación Técnica

Presentado por nombre alumno
en Universidad de Burgos — 17 de abril de 2024
Tutor: nombre tutor

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

La planificación meticulosa de proyectos de software se erige como un pilar fundamental para el éxito y la entrega oportuna de soluciones innovadoras. Este documento presenta el plan de proyecto para poder realizar un estudio sobre la deteccion y control de plagas.

El propósito de este plan es establecer un marco detallado que guíe las fases de concepción, diseño, implementación y despliegue del software. Se busca no solo asegurar la alineación con las necesidades del software y sus objetivos sino también maximizar la eficiencia de los procesos y la calidad del producto final. La planificación estratégica abordará componentes críticos como la definición de alcance, la gestión de riesgos, la asignación de recursos y los cronogramas de desarrollo.

A.2. Planificación temporal

Antes de comenzar con el desarrollo de proyecto,fueron planteadas las distintas posibles meotdologias que se utilizaran para la planificacion del desarrollo de este. Tras realizarse un estudio de cuales son las metodologias mas capables para este, se destacaron las siguientes:

- **Programación extrema**. Centrada en poenciar las relaciones interpersonales como clave para el éxito en el desarrollo software. [2]
- **Kanban**. Gestiona un óptimo flujo de trabajo dentro del proceso [5]

- **Scrum**. Proceso empírico, iterativo e incremental. [2]
- **Desarrollo Cascada**. Secuencia de fases, que alfinal de cada etapa reune toda la documentación. [4]

Llegando a la conclusión que la mas favorable a utilizar es Kanban debido a su buena representacion mediante el uso de tableros y a la mejora continua en cuanto al ajuste del trabajo de forma regular. Para la estructuración del trabajo utilizaremos:

- **Epics**. Esta es una historia de usuario de gran tamaño o alta granularidad y que tiene por lo tanto mayor grado de incertidumbre
- **Historias de usaurio**.Describen, en una o dos frases, una funcionalidad de software desde el punto de vista del usuario, con el lenguaje que éste emplearía [3]

En total fueron implementados 8 epics,los cuales fueron:

1. **Inicio del proyecto**. Este se enfocará en las tareas previas de organización y puesta en marcha del proyecto, como establecer los objetivos, la selección de metodologías y herramientas, y la asignación de recursos.
2. **Configuracion del entorno de desarrollo**. Este establece las actividades relacionadas con la configuración del entorno de desarrollo, asi mismo como la instalación del IDE, librerías y otras herramientas que necesitamos para el desarrollo del proyecto.
3. **Desarrollo del modelo**. Este epic se enfoca eb la creación y desarrollo del modelo de detección de insectos, incluyendo la implementación de algoritmos y técnicas necesarias para la correcta detección.
4. **Entrenamiento del modelo**. En este epic, se lleva a cabo el proceso de entrenamiento del modelo de detección de insectos, asi mismo como el establecimiento de los conjuntos de datos dedicados al entrenamiento, validación y pruebas.
5. **Observacion de los resultados**. Implica la evaluación y estudio de los resultados obtenidos por el modelo entrenado, analizando su precisión y certeza en la detección de los insectos.
6. **Afinamiento y ajuste**. En este epic, se realizan ajustes y mejoras en el modelo y en el proceso de detección de insectos, basados en los resultados obtenidos.

7. **Elaboracion web**. Este epic se enfoca en la creación y desarrollo de la interfaz web para el proyecto, que incluirá la visualización de resultados, la interacción con el usuario y otras funcionalidades relacionadas.

8. **Documentación**. Finalmente, este epic implica la elaboración de la documentación del proyecto.

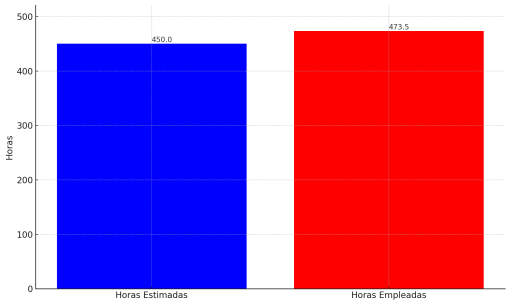

Figura A.1: Gráfico de horas empleadas

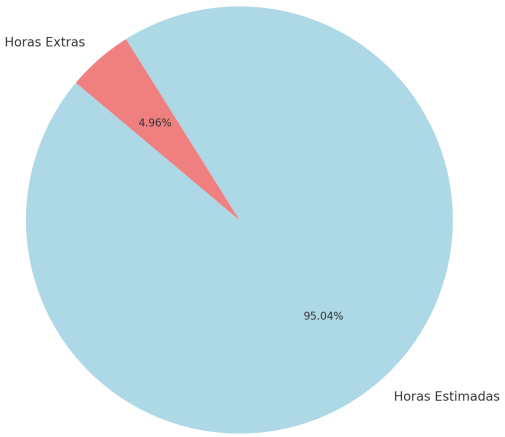

Figura A.2: Porcentajes de horas empleadas

En cuanto a estas horas empleadas en el proyecto estos son los siguientes porcentajes empleados en cada uno de los distintos campos del proyecto:

- 10 % planificacion previa.Engloba Inicio del proyecto y Configuracion del entorno de desarrollo

- 30 % elaboracion de desarrollo del modelo
- 10 % refinamiento modelo. Engloba Observacion de los resultados y Afinamiento.
- 15 % elaboracion web
- 35 % documentacion

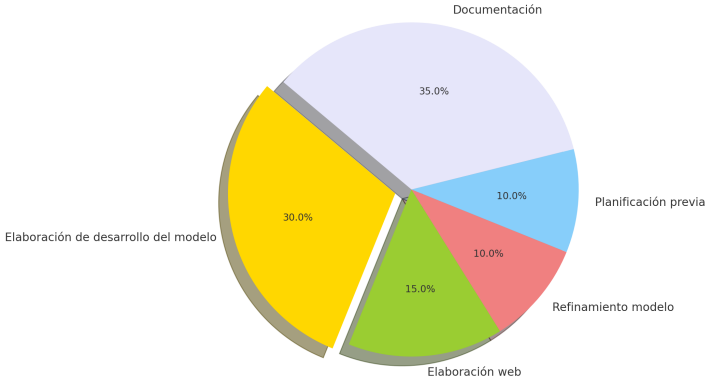

Figura A.3: Gráfico de horas empleadas en las distintas fases

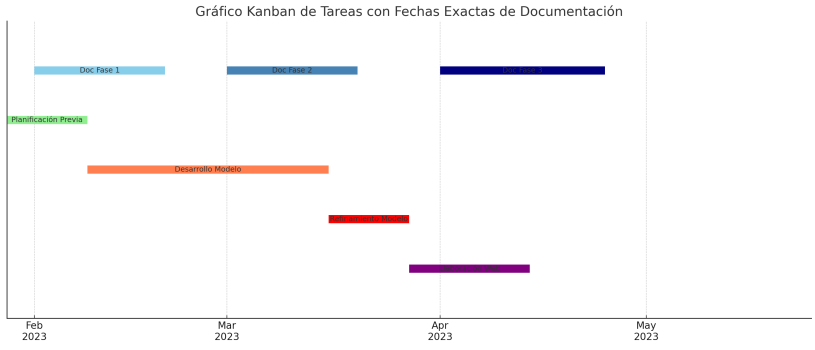

Figura A.4: Distribucion de las fases del proyecto
La documentación se creó en etapas, mientras que el desarrollo del modelo y afinamiento sin pausas.

A.3. Estudio de viabilidad

Viabilidad económica

La viabilidad económica es un factor determinante para la realización de este proyecto. Basándonos en ella, podremos determinar si el proyecto resultará exitoso o no, en función de si el coste total es superior o inferior al presupuesto estimado. Los costes del proyecto se clasificarán en tres categorías: directos, indirectos, fijos y variables.

Costes fijos Son costes que permanecen constantes a lo largo del desarrollo del proyecto, tales como:

- Alquiler del espacio de oficina.
- Salarios del personal.

Costes variables Estos costes fluctúan con el progreso del proyecto y pueden incluir:

- Estrategias de marketing y publicidad.
- Facturas de servicios como la electricidad, el agua y el gas.

Costes directos Son los costes que están vinculados directamente con las actividades del proyecto:

- Compra de trampas para insectos.
- Adquisición de cámaras y otros equipos de monitoreo.

Costes indirectos Estos costes no están relacionados directamente con las actividades diarias del proyecto pero son necesarios para su soporte:

- Mantenimiento y reparación de equipos.

Tabla A.1: Costes del proyecto.

CONCEPTO	CANTIDAD
Alquiler oficina	1200€
Salarios	5500,36€
Marketing	120€
Electricidad	312,54€
Agua	87,10€
Gas	95,02€
Trampas	80€
Cámaras	3200€
Mantenimiento de equipos	100€
TOTAL	**10695,02€**

Beneficios

El sistema de detección de insectos mediante IA y segmentación de imágenes ofrecerá una gama de beneficios en distintos niveles de aplicación y uso.

- **Básico:** Este plan es gratuito y satisface las necesidades básicas de los usuarios interesados en la identificación de insectos para aplicaciones no comerciales. Permite un número limitado de consultas al sistema de IA, ideal para pequeñas investigaciones o estudios de campo iniciales.
- **Profesional:** Orientado a usuarios que requieren un mayor volumen de análisis, como investigadores académicos o empresas agrícolas. Incluye un número mayor de consultas así como soporte técnico avanzado. El precio será ajustado en función de los requerimientos específicos.
- **Empresarial:** Para grandes corporaciones o proyectos de investigación intensiva que requieran un uso exhaustivo del servicio.

Viabilidad legal

Licencia del proyecto Desde el inicio de nuestro proyecto, nuestra intención ha sido ofrecer el software bajo una licencia de Software Libre. Hemos considerado varias licencias y hemos elegido la licencia MIT por su claridad y amplitud, lo que promueve la colaboración y uso extenso del software de código abierto.

La licencia MIT es ampliamente reconocida por su flexibilidad y mínimas restricciones sobre la redistribución, lo que está en línea con nuestra visión

de permitir un uso libre y amplio del proyecto por parte de la comunidad. Después de evaluar diversas opciones, concluimos que la licencia MIT era la más adecuada para nuestro proyecto, dada su simplicidad y la libertad que ofrece tanto a desarrolladores como a usuarios.

Optamos por la licencia MIT debido a su:

- **Permisividad:** Permite un uso extenso y variado del software, incluyendo aplicaciones comerciales.
- **Simplicidad:** Su texto breve y conciso evita complicaciones legales y es fácilmente entendible.
- **Fomento de la colaboración:** Es una licencia que anima a la mejora y contribución comunitaria.

Presentamos a continuación un resumen de los términos de la licencia MIT:

Permisos	**Limitaciones**	**Condiciones**
Uso comercial Modificación Distribución Uso privado	Sin responsabilidad Sin garantía	Aviso de licencia y derechos de autor

Tabla A.2: Resumen de la licencia MIT

La licencia completa se puede encontrar en el sitio web de Open Source Initiative en la siguiente URL: https://opensource.org/licenses/MIT.

Elegimos la licencia MIT porque:

- Facilita la difusión y utilización del software al no imponer restricciones significativas en la redistribución del código.
- Asegura que tanto colaboradores como usuarios puedan comprender los términos de la licencia de manera sencilla, fomentando así la colaboración.
- Refleja nuestro compromiso con la libertad de software y apoya un ecosistema de software abierto y colaborativo.

Apéndice B

Especificación de Requisitos

B.1. Introducción

Este apartado es esencial para establecer las expectativas claras y detalladas del proyecto, asegurando que todas las partes interesadas tengan un entendimiento común de los objetivos, funcionalidades y restricciones del sistema. Se centra en identificar y documentar:

- requisitos funcionales. especifican las propiedades y capacidades que debe tener el proyecto-
- no funcionales .especifican restricciones de calidad (por ej., usabilidad, mantenibilidad, etc.) sobre esas propiedades y capacidades.

La comprensión profunda de estos requisitos es fundamental para guiar el diseño, desarrollo y pruebas posteriores, facilitando así la creación de una solución efectiva y eficiente para el desarrollo del proyecto

B.2. Objetivos generales

Los objetivos generales del proyecto a desarrollar son los siguientes:

- Elaborar una aplicación web que permita el usuario interactuar con nuestro modelo de detección de insectos de forma intuitva.
- Desarrollar un modelo de detección de insectos que permita el reconocimiento de estos mediante el uso de *redes neuronales* y *Mask-Method MRCNN*

- Dar a entender al usuario un conocimiento sobre el proceso del algoritmo y sobre *sejmentacion de imagenes*
- Aprender sobre el entrenamiento de modelos a traves del lenguaje de programacion *Python* y de sus librerias especializadas *Tensorflow* y *openCV*

B.3. Catálogo de requisitos

Requisitos funcionales

- **RF-1: Carga de Imágenes**: El usuario debe ser capaz de cargar imágenes en el sistema para su análisis.
  - **RF-1.1: Carga para Entrenamiento**: Permite subir imágenes al conjunto de entrenamiento del modelo de detección.
  - **RF-1.2: Carga para Predicción**: Facilita la subida de nuevas imágenes para realizar predicciones inmediatas.
- **RF-2: Procesamiento de Imágenes**: El sistema procesará las imágenes utilizando el algoritmo de IA.
  - **RF-2.1: Preprocesamiento**: Aplicar técnicas de preprocesamiento para preparar las imágenes para el modelo.
  - **RF-2.2: Segmentación de Imágenes**: Ejecutar el algoritmo Mask-Method MRCNN para segmentar los insectos en las imágenes.
- **RF-3: Interfaz de Usuario Intuitiva**: Proporcionar una GUI que permita a los usuarios interactuar fácilmente con el sistema.
- **RF-4: Visualización de Resultados**: Mostrar los resultados de la detección de manera que los usuarios puedan comprender fácilmente.
- **RF-5: Educación de Usuario**: Incluir una sección educativa que explique los conceptos de segmentación de imágenes y redes neuronales.

Requisitos no funcionales

- **RNF-1: Usabilidad**: La aplicación web debe ser intuitiva y de fácil navegación para usuarios de todos los niveles técnicos asi mismo no superar los 10 minutos en aprender a usar la aplicación.

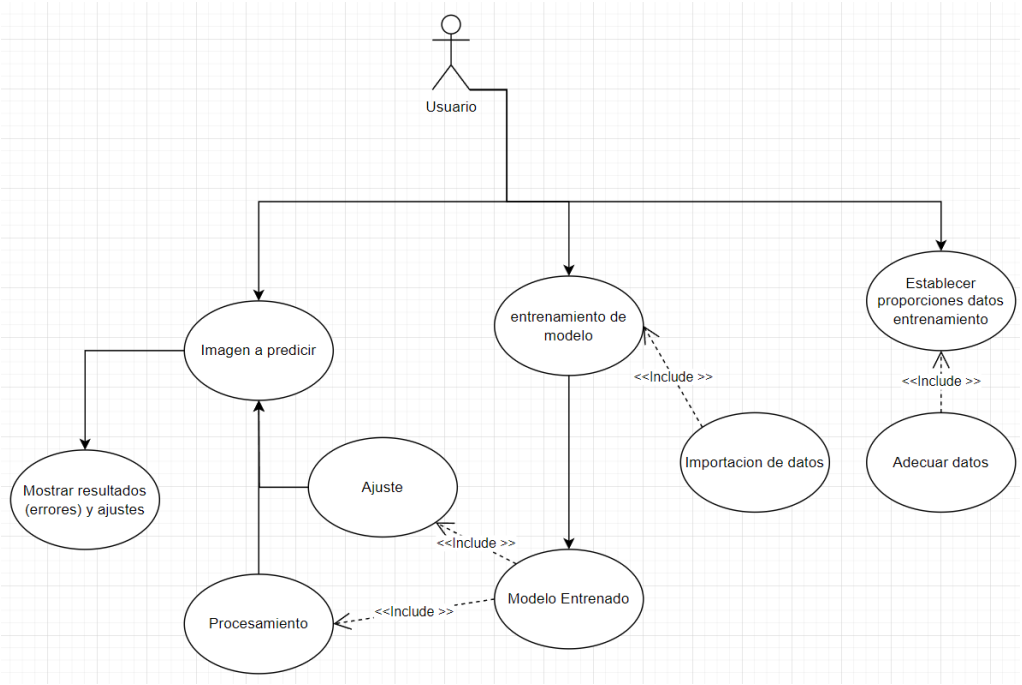

Figura B.1: Diagrama de casos de uso

- **RNF-2: Rendimiento**: El modelo de detección debe realizar predicciones en un tiempo de respuesta rápido.
- **RNF-3: Escalabilidad**: El sistema debe ser capaz de escalar para manejar un aumento en la carga de trabajo sin pérdida de rendimiento.
- **RNF-4: Seguridad**: Implementar protocolos de seguridad para la protección de datos y privacidad del usuario.

B.4. Especificación de requisitos

CU-1	**Adecuar Datos**
Versión	1.0
Autor	Arturo Carretero Mateo
Requisitos asociados	RF-2.1
Descripción	Preparar el formato de las imagenes y sus equivalentes etiquetados posteriormente guardandolos en el path correspondiente .
Precondición	Imágenes cargadas en el sistema listas para ser procesadas.
Acciones	1. Renombra las imagenes con el nombre 000n donde n es el numero de la imagen (esto para poder referenciarlas posteriormente) 2. Asigna las imagnes a su correspondiente etiquetado 3. Establece las imagenes en el path correspondiente 4. Desordena las imagenes y su etiquetado(con el fin de ser diferente orden en los entrenamientos).
Postcondición	Las imagenes se encuentran en el correcto path para su posterior cargar.
Excepciones	Excepcion al no corresponder etiqueta y imagen. Excepcion al renombrar una imagen
Importancia	Alta

Tabla B.1: CU-1 Adecuar Datos.

CU-2	**Carga de Imágenes**
Versión	1.0
Autor	Arturo Carretero Mateo
Requisitos asociados	RF-1, RF-1.1, RF-1.2
Descripción	Permitir al usuario cargar imágenes al sistema para su análisis y procesamiento.
Precondición	Los datos han de tener una extension y nombres correctos.
Acciones	1. Comprobacion de formato correcto de imagenes. 2. Se ordenan con su correspondiente etiquetado. 3. Se guardan en el path concreto. 4. Las imágenes se cargan en el sistema para el proceso seleccionado.
Postcondición	Los .jpg han de corresponder con sus correspondientes etiquetados .xml
Excepciones	Mal etiquetado Nula correspondencia Mal formato
Importancia	Alta

Tabla B.2: CU-2 Carga de Imágenes.

CU-3	**Establecer Proporciones de Datos de Entrenamiento**
Versión	1.0
Autor	Arturo Carretero Mateo
Requisitos asociados	RF-1.1, RF-2.1
Descripción	Definir y aplicar las proporciones adecuadas entre los conjuntos de datos de entrenamiento, validación y prueba para optimizar el rendimiento del modelo de IA.
Precondición	Los datos han sido adecuados y están listos para ser divididos en los diferentes conjuntos.
Acciones	1. Determinar las proporciones óptimas para el conjunto de datos basándose en las mejores prácticas y en la cantidad de datos disponibles. 2. Dividir el conjunto de datos total en entrenamiento, validación y prueba según las proporciones establecidas. 3. Asegurarse de que la distribución de las clases sea homogénea en cada subconjunto para evitar sesgos en el entrenamiento. 4. Guardar los conjuntos de datos en ubicaciones separadas, claramente etiquetadas para su uso durante el entrenamiento y la evaluación del modelo.
Postcondición	Los conjuntos de datos de entrenamiento, validación y prueba están preparados y distribuidos adecuadamente, listos para ser utilizados en el proceso de entrenamiento y evaluación del modelo.
Excepciones	Cantidad insuficiente de datos que impide una división equitativa, distribución desigual de las clases en los subconjuntos.
Importancia	Alta

Tabla B.3: CU-3 Establecer Proporciones de Datos de Entrenamiento.

CU-4	**Entrenamiento del Modelo**
Versión	1.0
Autor	Arturo Carretero Mateo
Requisitos asociados	RF-1.1, RF-2.2
Descripción	Realizar el proceso de entrenamiento del modelo de IA para la detección y segmentación de insectos.
Precondición	Conjunto de datos de entrenamiento cargado y preprocesado.
Acciones	1. Preparar los datos de entrenamiento y validar su formato y calidad. 2. Configurar los parámetros del modelo y el entorno de entrenamiento. 3. Iniciar el proceso de entrenamiento del modelo con los datos disponibles. 4. Evaluar el rendimiento del modelo utilizando un conjunto de datos de validación. 5. Ajustar los parámetros del modelo si es necesario y repetir el entrenamiento.
Postcondición	El modelo ha sido entrenado y está listo para ser evaluado o utilizado en producción generando asi un modelo nuevo. El modelo guardado en la correspondiente dirección
Excepciones	Problemas durante el entrenamiento como overfitting, underfitting, o errores en los datos.
Importancia	Alta

Tabla B.4: CU-4 Entrenamiento del Modelo.

CU-5	**Procesamiento de Imágenes**
Versión	1.0
Autor	Arturo Carretero Mateo
Requisitos asociados	RF-2, RF-2.1, RF-2.2
Descripción	Procesar las imágenes cargadas utilizando algoritmos de IA para detectar y segmentar insectos.
Precondición	Imágenes cargadas y listas para procesar.
Acciones	1. El sistema aplica técnicas de preprocesamiento a las imágenes. 2. El sistema ejecuta el algoritmo Mask-Method MRCNN para la segmentación. 3. El sistema guarda los resultados del procesamiento.
Postcondición	Imágenes procesadas con insectos segmentados disponibles para revisión.
Excepciones	Fallas en el procesamiento debido a errores en los datos o en los algoritmos.
Importancia	Alta

Tabla B.5: CU-5 Procesamiento de Imágenes.

CU-6	**Predicción de Imagen por el Modelo**
Versión	1.0
Autor	Arturo Carretero Mateo
Requisitos asociados	RF-1.2, RF-4
Descripción	Permitir al usuario introducir una imagen en el sistema para que el modelo de IA realice una predicción, identificando y segmentando los insectos presentes.
Precondición	El modelo de IA ha sido entrenado adecuadamente y está disponible para realizar predicciones.
Acciones	1. El usuario sube una imagen al sistema a través de la interfaz de usuario. 2. El sistema verifica el formato y la calidad de la imagen. 3. La imagen se procesa utilizando el modelo de IA para detectar y segmentar los insectos. 4. Los resultados de la predicción se muestran al usuario, incluyendo la localización y clasificación de los insectos detectados. 5. Se ofrece la opción de guardar los resultados o realizar ajustes si el sistema lo permite.
Postcondición	El usuario recibe los resultados de la predicción, incluyendo la identificación y segmentación de los insectos en la imagen.
Excepciones	Imagen de baja calidad que impide la detección, formato de imagen incompatible, errores en el modelo de IA.
Importancia	Alta

Tabla B.6: CU-6 Predicción de Imagen por el Modelo.

CU-7	**Mostrar Resultados y Ajustes**
Versión	1.0
Autor	Arturo Carretero Mateo
Requisitos asociados	RF-2, RF-4
Descripción	Visualizar los resultados de la detección y segmentación de insectos, incluyendo la identificación de errores y la posibilidad de realizar ajustes.
Precondición	La imagen ha sido procesada y analizada por el modelo.
Acciones	1. Presentar los resultados de la detección de insectos en la interfaz de usuario. 2. Destacar errores o áreas de incertidumbre en los resultados. 3. Ofrecer opciones para ajustar la detección o para reentrenar el modelo con nuevos datos. 4. Guardar los ajustes realizados por el usuario para futuras predicciones.
Postcondición	Los resultados se muestran correctamente y los ajustes quedan registrados en el sistema.
Excepciones	Fallas en la visualización de resultados
Importancia	Alta

Tabla B.7: CU-7 Mostrar Resultados y Ajustes.

Apéndice C

Especificación de diseño

C.1. Introducción

En esta sección se desarrollaran las diferenetes elecciones sobre el diseño realizado para el sistema siendo estas: diseño de datos, diseño procedimental y diseño arquitectónico.

Centrandonos en fijar los determinados objetivos los cuales se han de seguir para el desarrollo del proyecto

C.2. Diseño de datos

Para la obtencion de datos he utilizado 4TU.ResearchData

4TU.ResearchData Este es un repositorio internacional de datos para ciencia, ingeniería y diseño. Ofrece servicios de curaduría, compartición, acceso a largo plazo y conservación de conjuntos de datos de investigación, disponibles para cualquier persona en el mundo. También brinda formación y recursos para ayudar a los investigadores a hacer que los datos de investigación sean localizables, accesibles, interoperables y reproducibles. Alberga miles de conjuntos de datos completos y valiosos de investigaciones técnico-científicas. Estos pueden ser datos crudos, código, datos procesados o datos específicos . [6]

En el repositorio se ha decidido utilizar el datasheet de: Raw data from Yellow Sticky Traps with insects for training of deep learning Convolutional Neural Network .Ya que la base de datos contiene imágenes etiquetadas de

insectos recolectados en invernaderos comerciales, lo cual es fundamental para entrenar algoritmos de aprendizaje profundo en tareas de reconocimiento y clasificación. Además, está disponible bajo una licencia CC0 que permite su uso sin restricciones de derechos de autor, lo que facilita la investigación y el desarrollo en el campo de la inteligencia artificial.

Conjunto de datos En cuanto a el uso de los datos durante el proyecto, destacan 2 fase:

1. Antes del entrenamiento (Pre-entrenamiento)
2. Despues del entrenamiento (Post-entrenamiento)

Antes del entrenamiento Utilizaremos un conjunto de pesos preentrenados, este tiene el nombre de mask_rcnn_coco.h5 utiliza Mask R-CNN que y ha ha sido entrenado utilizando el dataset MS COCO. Este dataset es ampliamente utilizado para tareas de detección de objetos, segmentación y generación de subtítulos para imágenes. Los pesos contenidos en este archivo corresponden a la configuración de la red neuronal después de haber sido entrenada con dicho dataset, y se utilizan para inicializar el modelo Mask R-CNN para entrenamientos adicionales o para realizar predicciones directamente

Este archivo de pesos nos servirá como punto de partida, aprovechando el aprendizaje transferido para adaptar el modelo preentrenado a un nuevo conjunto de datos y asi no tener que empezar desde un archivo de pesos desde 0.

Después del entrenamiento

C.3. Diseño procedimental

C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

D.1. Introducción

En este apéndice se presenta la documentación esencial para llevar a cabo la correcta programación del proyecto. Esta profundiza sobre la estructura y disposición necesaria para el funcionamiento del proyecto,además del manual del programador para comprender mejor la programación de este. De igual modo, menciona el proceso que se lleva acabo a la hora de la instalacion y ejecución de este y sus pruebas correspondientes.

D.2. Estructura de directorios

PILLAR DRAWIO Y FOTO DE LA ESTRUCTURA DE DIRECTORIOS HABLAR POR ENCIMA DE ESTOS

D.3. Manual del programador

En esta sección se especficará los disitntos procesos que se siguieron durante la isntalción. El leector (con un nivel medio de programación) ha de ser capaz de seguirla e incluso ampliarla pudiendo llegar a presentar una mejora de este.

Requisitos de Hardware Antes de comenzar con la instalación y desarrollo del proyecto, es importante asegurarse de que tu equipo cumple con los siguientes requisitos de hardware:

- **Procesador**: Se recomienda tener un procesador con múltiples núcleos y capacidad para ejecutar tareas intensivas en cómputo.
- **Memoria RAM**: Se recomienda tener al menos 8 GB de memoria RAM, aunque más memoria RAM puede ser beneficiosa para trabajos más grandes y complejos.
- **Tarjeta Gráfica NVIDIA (GPU)**: Se requiere una tarjeta gráfica NVIDIA con soporte CUDA para aprovechar al máximo las capacidades de procesamiento paralelo y acelerar el entrenamiento de modelos de aprendizaje profundo.

Requisitos de Software Además de los requisitos de hardware, también necesitarás cierto software instalado en tu sistema para poder desarrollar y ejecutar el proyecto correctamente:

- **Sistema Operativo**: Se recomienda tener un sistema operativo compatible con las herramientas y bibliotecas utilizadas en el proyecto en mi caso he utilizado Windows 10 arquitecura de 64 bits.
- **Python**: Python es un requisito fundamental para el desarrollo del proyecto. Asegúrate de tener instalada una versión de Python compatible, preferiblemente Python 3.7.11 o superior, mas adelante se especifica sobre su instalación.
- **CUDA Toolkit**: Si planeas utilizar la GPU para el entrenamiento de modelos de aprendizaje profundo, necesitarás instalar CUDA Toolkit en tu sistema, mas adelante se especifica sobre su instalación.

Instalación

Para poder desarollar este proyecto se ha de tener previamente instalado *Python*. Para poder instalarlo acudiremos al sigueinte enlace:
https://www.python.org/downloads.
Seleccionaremos la version mas apropiada descargandose posteriormente el instalador siguiendo los pasos recomendados posteriormente.
Este paso puede ser crucial ya que hay algunas librerias que utilizaremos

posteriomente pueden dar problemas de compatibilidad unas y otras , en mi caso contaré con la version de python 3.7.11.
Para consultar tu version instalada comprobarla con el comnado:
python –version

Utilizaremos la GPU el máximo de sus capacidades para el entrenamiento para ello instalremos CUDA en nuestro equipo, permitindo asi el paralelismo. Para la instlacion de este nos tendremos previamente que asegurarnos de tener una tarjeta gráfica de NVIDIA y el sisema operativo ser compatible. Para instalar CUDA toolkit iremos al siguiente enlace: https://developer.nvidia.com/cuda-toolkit.

Para la instalación de Mask R-CNN utilizare un script el cual tiene el nombre de *setup.py* y con el comando *python setup.py install* podremos reaizar la instalación sin problemas.

Además de implementar Mask R-CNN, CUDA y Python, me gustaría resaltar el empleo de GitHub en este procedimiento para administrar el código fuente de nuestro proyecto. GitHub nos proporciona una plataforma para el control de versiones distribuido, permitiéndonos mantener un registro de todos los cambios realizados en nuestro código.
Para acceder a este, debemos acudir al siguiente enlace:
https://github.com/arturo1026/Deteccion_Insectos_TFG.

Entorno para la programación

Para la administración del entorno y de las dependencias utilizaremos Anaconda, el cual es una distribucion muy utilizada en ciencia de datos, y aprendizaje automático incluyendo procesamiento de grandes volúmenes de información, análisis predictivo y cómputos científicos. [1]

Para poder descargar la distribución acudiremos al siguiente enlace: https://www.anaconda.com/products/distribution.

Una vez que lo hemos instalado y dentro de este tendremos la siguiente vista:

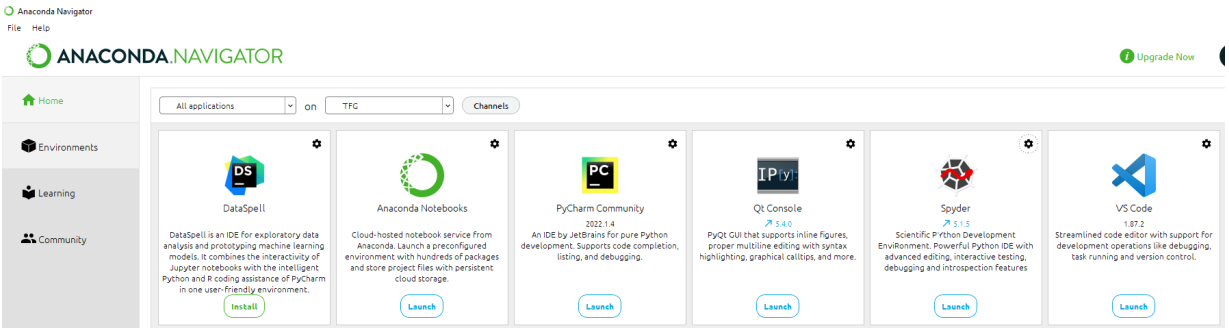

Figura D.1: Vista por defecto anaconda

En mi caso utilizaré el IDE Spyder con una versian 5.1.5

Funciones y clases

- **load_dataset(dataset_dir, is_train=True)**: Esta función carga el conjunto de datos de insectos. Definiendo las clases disponibles en el conjunto de datos y sus ubicaciones. Recorre todas las imágenes en el directorio de imágenes, extrayendo el ID de la imagen y verificando si es parte del conjunto de entrenamiento o de prueba. Luego, agrega la imagen al conjunto de datos junto con su anotación.

- **extract_boxes(filename)**: Esta función extrae las coordenadas de las cajas delimitadoras de una anotación XML dada. Lee el archivo XML y extrae las coordenadas de las cajas delimitadoras de los objetos presentes en la imagen.

- **load_mask(image_id)**: Esta función carga las máscaras correspondientes a una imagen dada. Obtiene los detalles de la imagen a partir de su ID, extrae las cajas delimitadoras y genera las máscaras correspondientes.

- **image_reference(image_id)**: Esta función devuelve la ruta de la imagen correspondiente a un ID de imagen dado.

- **InsectConfig(Config)**: Esta CLASE define la configuración del modelo de Mask R-CNN para el conjunto de datos de insectos. Define el nombre de la configuración, el número de clases, y los pasos por época durante el entrenamiento.

- **model.train(train_set, test_set, learning_rate=config.LEARNING_RATE, epochs=x, layers='heads')**: Esta función entrena el modelo de Mask R-CNN utilizando los conjuntos de datos de entrenamiento y prueba especificados. Configura el modelo con la configuración proporcionada y entrena el modelo durante el número especificado de épocas(epochs).

- **PredictionConfig(Config)**: Esta CLASE define la configuración para realizar predicciones utilizando el modelo entrenado. Especifica el nombre de la configuración y el número de clases, entre otros detalles.

- **Rename_file_to_numbers**:renombra todos los archivos en una carpeta a números secuenciales. Utiliza módulos como os, glob y random para interactuar con el sistema de archivos, buscar archivos y generar números aleatorios. Luego, aleatoriza la lista de archivos, asigna nombres secuenciales y los renombra en orden ascendente.

D.4. Compilación, instalación y ejecución del proyecto

El lenguaje que utilizaremos es Python por lo tanto, se trata de un lenguaje interpretado y no compilado sin generarse ningun tipo de ejecutable. Como tal interpretaremos los programas atraves del IDE especficado anteriormente y teniendo python instalado. Para la instalacion de las dependencias tendremos que instalarlas mediante el archivo requeriments.txt que se encuentra en el raiz del proyecto.

- Instalaremos con el comando: *pip install -r requirements.txt*

- Comprobamos la correcta instalación con el comando: *pip list*

Con estos pasos, podemos asegurar la correcta instalacion y configuración de las dependencias para la ejecución del proyecto de manera correcta.

D.5. Pruebas del sistema

INCLUIR ASSERTS Y PRUEBAS
Podremos ir probando las distintas partes de nuestro proyecto ya que al utilizar Spyder nos permitira ejecutar una porción de codigo que nostros hayamos seleccionado o bie podremos agrupar el codigo mediante celdas e ir ejecutandolas.
Ademas de poder fragmentar el codigo y ir comprobandolo podremos utilizar el Debugger de este ,comprobando asi el flujo de la ejecución y como cambian las variables a lo largo de la ejecución de este.

Apéndice E

Documentación de usuario

E.1. Introducción

En este apartado, se abordan los aspectos esenciales relacionados con los requisitos y procedimientos necesarios para la correcta ejecución y uso del programa desarrollado. Se detallan tanto los requisitos que la aplicación demanda, como las instrucciones para su instalación y utilización por parte del usuario final.

E.2. Requisitos de usuarios

Tendremos que contar con un equipo, ya sea portátil o de escritorio, para llevar a cabo el proyecto informático. Además, será necesario cumplir con los requisitos mínimos tanto de software como de hardware que Python, Spyder y otras herramientas asociadas al proyecto requieran. Esto incluye tener instalado:

- Sistema operatio: Windows XP - Windows Server 20008 - Windows 2003 o versiones superiores.
- Se recomienda tener suficiente almanceamiento de disco duro para el almacenamiento del dataset y la flexilibidad de aumento de este
- Debe de tener como mínimo al menos 8 GB de memoria RAM.
- Permisos de administrador para poder ejecutar el programa.
- Un navegador para poder acceder a la aplicacion Flask.

E.3. Instalación

En caso de ser necesario se recomienda ver las instrucciones para la instalacion de python y de Spyder en el apartado de D.3

Ejecucion por Prompt

Ejecucion por aplicacion

E.4. Manual del usuario

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

Este anexo incluirá una reflexión personal del alumnado sobre los aspectos de la sostenibilidad que se abordan en el trabajo. Se pueden incluir tantas subsecciones como sean necesarias con la intención de explicar las competencias de sostenibilidad adquiridas durante el alumnado y aplicadas al Trabajo de Fin de Grado.

Más información en el documento de la CRUE https://www.crue.org/wp-content/uploads/2020/02/Directrices_Sosteniblidad_Crue2012.pdf.

Este anexo tendrá una extensión comprendida entre 600 y 800 palabras.

Bibliografía

[1] Anaconda (distribución de python). https://es.wikipedia.org/wiki/Anaconda_(distribuci%C3%B3n_de_Python). Consultado el día Mes, Año.

[2] Sarah Damaris Amaro Calderon and Jorge Carlos Valverde Rebaza. Metodologías ágiles. *Universidad Nacional de Trujillo*, 37, 2007.

[3] Alexander Menzinsky, Gertrudis López, Juan Palacio, M Sobrino, Rubén Álvarez, and Verónica Rivas. Historias de usuario. *Ingeniería de requisitos ágil*, 2018.

[4] B Molina Montero, H Vite Cevallos, and J Dávila Cuesta. Metodologias agiles frente a las tradicionales en el proceso de desarrollo de software. *Espirales revista multidisciplinaria de investigación*, 2(17):114–121, 2018.

[5] Hubner Janampa Patilla, Edgar Gómez Enciso, José Carlos Juárez Pulache, Jorge Luis Lozano Rodríguez, Eder Solórzano Huallanca, and Yudith Meneses Conislla. Modelo de gestión de desarrollo de software ágil mediante scrum y kanban sobre la programación extrema. *Revista Ibérica de Sistemas e Tecnologias de Informação*, (E43):450–466, 2021.

[6] TU Delft Library. 4tu.researchdata - TU Delft Library, 2024. Último acceso el 17 de Marzo de 2024.