Instituto Tecnológico de Costa Rica

Carrera de Ingeniería en Computación,
Campus Tecnológico Local San Carlos

"Aplicación móvil Android para la detección de las enfermedades *Fusarium* y *Phytophtora* en cultivos de piña"

Miguel Adán Rivas Méndez

Costa Rica, 2021

1. Resumen ejecutivo

Este proyecto tiene como propósito el desarrollo de una aplicación móvil para la detección de la enfermedades *Fusarium* y *Phytophtora* utilizando modelos de aprendizaje automático de Redes Neuronales Convolucionales, este aplicación tiene como fin hacer posible la detección de dichas enfermedades en poco tiempo a través de experimentos capaces de realizarse en el campo.

2. Palabras clave

android, *Fusarium* , *Phytophtora* , *machine learning*, piña, redes neuronales convolucionales

Índice general

1. Resumen ejecutivo 1
2. Palabras clave 1

1. Capítulo 1 **6**
1. Introducción 6
   1.1. Descripción de empresa 6
   1.2. Contexto del proyecto 6
   1.3. Problema 7
   1.4. Objetivos 7
   1.5. Justificación 7

2. Capítulo 2 **9**
2. Revisión de literatura 9
   2.1. Marco teórico 9
   2.2. Trabajos relacionados 11

3. Capítulo 3 **13**
3. Solución planteada 13
   3.1. Propuesta 13
   3.2. Involucrados 14
   3.3. Procedimiento metodológico 15
   3.4. Análisis de los riesgos 16
   3.5. Cronograma de trabajo 16

4. Capítulo 4 **17**
4. Requerimientos y Diseño 17
   4.1. Definición de requerimientos 17
   4.2. Diseño de la plataforma de software 19

5. Capítulo 5 **21**
5. Plataforma de software 21
   5.1. Aplicación 21
   5.2. Evaluación 27

6. Capítulo 6 **31**
6. Conclusiones 31
6.1. Recomendaciones 31

Índice de figuras

2.1. Flujo básico de una Red Neuronal Convolucional. 11

3.1. Cronograma de trabajo del proyecto. 16

4.1. Arquitectura de la aplicación móvil. 20

5.1. Arquitectura de la aplicación móvil. 24
5.2. Vista principal de la aplicación Fusapp 25
5.3. Arquitectura de componentes de la aplicación móvil. 26
5.4. Algunas vistas de la aplicación. 26
5.5. Métricas de Precisión, Exhaustividad *(Recall)* del modelo. 27
5.6. Métrica de F1. 28
5.7. Consumo de CPU de Fusapp durante operaciones relacionadas al modelo móvil. 29
5.8. Consumo de memoria de Fusapp durante operaciones relacionadas al modelo móvil. 30
5.9. Consumo de energía de Fusapp durante operaciones relacionadas al modelo móvil. 30

Índice de cuadros

3.1. Lista de involucrados 14
3.2. Matriz para el procedimiento metodológico 15
3.3. Análisis de riesgos 16

4.1. Requerimientos del proyecto 19

5.1. Parámetros de entrenamiento del modelo *Fusapp* 22
5.2. Especificaciones de máquina virtual utilizada para desplegar servidor. . . . 28

Capítulo 1

1. Introducción

En este documento se registra el proceso de elaboración del proyecto **"Aplicación móvil Android para la detección de las enfermedades *Fusarium* y *Phytophtora* en cultivos de piña"**, un trabajo enfocado a la creación de herramientas para la detección de enfermedades en la piña, a través del uso de técnicas de aprendizaje automático y la plataforma de desarrollo Android para la entrega de una aplicación móvil.

1.1. Descripción de empresa

El Tecnológico de Costa Rica (TEC) es una institución nacional autónoma de educación superior universitaria, dedicada a la docencia, la investigación y la extensión de la tecnología y las ciencias conexas para el desarrollo de Costa Rica.

El desarrollo de este proyecto interno se llevará a cabo junto al Tecnológico de Costa Rica, concretamente en el Campus Tecnológico Local San Carlos, esto dentro del esquema de investigación del laboratorio de PROTEC, en conjunto con la mentoría del profesor Rogelio González Quirós.

1.2. Contexto del proyecto

Con el desarrollo de la computación en el campo del aprendizaje automático, se ha conseguido elaborar herramientas que facilitan tareas en un sinfín de áreas ajenas a la computación. Una de estas áreas es la agricultura, concretamente en la detección de enfermedades. Esta es una tarea que requiere de técnicas de laboratorio para llevarse a cabo, y que representan un costo.

Gracias a las bondades de la visión por computadora y el aprendizaje automático, es posible agilizar la detección patrones que requieren mucho tiempo por parte de una persona.

En este proyecto nos enfocaremos en la detección de enfermedades en cultivos de piña, a través de los beneficios de las Redes Neuronales Convolucionales, aplicando técnicas de segmentación semántica, para luego, usar dicha detección en una aplicación móvil para Android.

1.3. Problema

La detección de enfermedades en plantas es un proceso que requiere tiempo y debido a su naturaleza, además, se necesita recurso humano capacitado y es un proceso repetitivo basado en la búsqueda de patrones.

Hay una importante presencia de productores de piña en la Zona Norte, un cultivo que no está excluido de sufrir una variedad de enfermedades, entre ellas el *Fusariosis de la piña*, una enfermedad causada por un hongo, que causa considerables pérdidas económicas a dichos productores.

Existe la necesidad de técnicas de identificación temprana de patologías en plantas, debido a que actualmente se requiere disponer de laboratorios y expertos para el estudio y detección de dichas enfermedades.

1.4. Objetivos

Objetivo general

Desarrollar una aplicación móvil en dispositivos Android para la detección de la enfermedades *Fusarium* y *Phytophtora* en cultivos de piña para los productores de la Región Huertar Norte por medio de aprendizaje automático.

Objetivos específicos

1. Recopilar un conjunto de datos de fotografías de piña etiquetadas para alimentar un modelo de aprendizaje automático de detección de enfermedades.

2. Diseñar un modelo de aprendizaje automático que permita detectar la enfermedades *Fusarium* y *Phytophtora* en las plantas de piñas a través de fotografías etiquetadas con segmentación semántica.

3. Desarrollar una aplicación móvil nativa Android para la detección de la enfermedades *Fusarium* y *Phytophtora* en las plantas de piña.

1.5. Justificación

Este proyecto tiene como fin la detección de la enfermedad *Fusarium* en la planta de piña a través del uso de aprendizaje automático y una aplicación móvil. El producto

innovador resultante permitirá que sea posible realizar dicha tarea de forma rápida, en el campo y sin la necesidad de las tareas de análisis en laboratorio tradicionales, lo que permite que optimizar los procesos de detección de la enfermedad, así como también la reducción de costos.

La realización de este proyecto representa un beneficio para los productores de piña de la Zona Norte, cultivo que representa una fracción importante de la economía de la región, con un 49 % de toda la producción del país [5].

Capítulo 2

2. Revisión de literatura

En este capítulo se recopilan las bases literarias sobre las que se fundamenta el proyecto, se incluyen conceptos e ideas necesarias para abordar el alcance del proyecto, así como también, trabajos relacionados de impacto.

2.1. Marco teórico

En esta sección se describen las bases teóricas necesarias para el desarollo del presente proyecto.

Cultivo de piña

Actualmente, el cultivo de piña constituye una importante economía en Costa Rica, en 2019, se realizaron exportaciones por $930.49 M [5].

Sin embargo, el cultivo de esta planta requiere prevención de una variedad de amenazas de diferente origen. De hecho, los hongos son unos de los organismos que representan un problema en estos cultivos, son responsables de una amplia gama de enfermedades. Por ello, en este caso, nos enfocaremos en la detección de los hongos *Fusarium* y *Phytophtora* .

- ***Fusarium*** . El género Fusarium es un grupo de hongos filamentosos ampliamente distribuidos en el suelo y plantas. Debido a su capacidad de crecer a 37°C, son considerados oportunistas. Pueden causar infecciones sistémicas en pacientes inmunocomprometidos, con una alta mortalidad. Algunas de sus especies producen toxinas que afectan al hombre y animales. [12] [11]

- ***Phytophtora*** . Es un patógeno que provoca la marchitez de decadencia de frutos y raíces de muchas especies de cultivos comerciales importantes. Además, este patógeno puede sobrevivir durante varios años en el suelo, principalmente en las capas superficiales. Las altas temperaturas y el exceso de agua en el suelo favorecen su desarrollo. [4]

Desarrollo móvil

La modernización de los dispositivos móviles ha hecho posible realizar tareas cada vez más complejas, en el escenario de este proyecto, el desarrollo de una aplicación móvil Android para dispositivos competentes, permitirá la detección de la enfermedad *Fusarium* a partir de las facilidades que provee el *Software Development Kit* (SDK) de Android, que es el conjunto de herramientas que este sistema provee para desarrollar soluciones para la plataforma. En concreto, el SDK de Android permite desarrollar en los lenguajes de programación *Java* y *Kotlin*, además de C++ para aplicaciones de altas exigencias. Para este proyecto se hace uso de Kotlin.

Aprendizaje automático

El Aprendizaje Automático refiere a soluciones de las Ciencias de la Computación diseñadas para implementar el aprendizaje en los sistemas informáticos, a través de Teoría de Álgebra Lineal, Probabilidad y Estadística, y otros campos de la matemática, cómo el Cálculo Vectorial. Además, se interconecta con otras disciplinas externas para brindar soluciones relacionadas.

A pesar de que es un campo muy amplio, para el ámbito del proyecto nos enfocaremos en el tema de las Redes Neuronales, específicamente, en las **Redes Neuronales Convolucionales**, en adelante **CNN**'s.

- **Red Neuronal.** Es un modelo computacional de tipo grafo, donde los vértices representan las unidades denominadas neuronas, las aristas representan las conexiones con otras neuronas. Estos modelos se encuentran estructurados en capas,usualmente se tiene una capa de entrada, una o más intermedias u ocultas, y una capa de salida, cada capa puede tener varias neuronas. El nombre de este modelo se debe a que el objetivo es resolver problemas de la misma forma que sucede en el cerebro humano.

  Las neuronas tienen asociadas funciones que le permiten validar los datos de entrada, existen diferentes tipos de estas funciones: de coste, de activación, por ejemplo.

- **Red Neuronal Convolucional.** Son un tipo de red neuronal enfocada en imitar el comportamiento de las neuronas en la corteza visual primaria, esto hace que sean la opción ideal para la extracción de información en imágenes. Es por ello que son de naturaleza multicapa y multidimensional.

  Una neurona CNN está diseñada para operar sobre matrices, debido a que recibe imágenes 2D.

  En la figura 2.1 se puede observar un diagrama que resume de forma muy simple el flujo de una Red Neuronal Convolucional, donde,

  1. Se da una imagen como dato de entrada, esta imágen puede estar compuesta de uno o múltiples canáles de color.

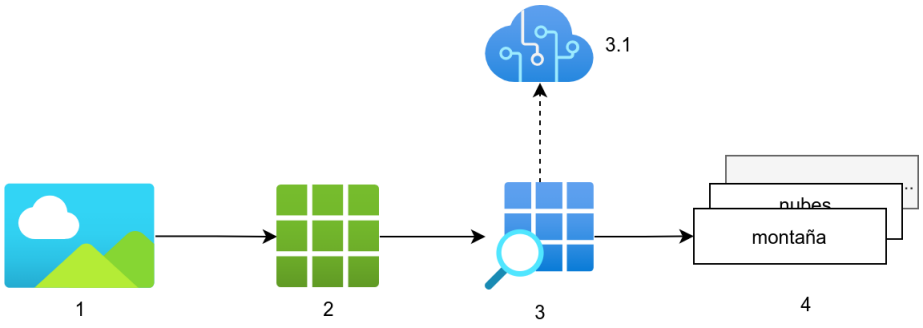

Figura 2.1: Flujo básico de una Red Neuronal Convolucional.

2. La imagen se convierte a una matriz de valores numéricos, correspondientes a los píxeles de la imagen. Si la imagen tiene más de un canal de color, se generará una matriz por cada uno de estos, por ejemplo, si la imagen tiene usa RGB, se generarán tres matrices, para cada color.
3. Se analiza la imagen a través de modelos matemáticos basados en operadores convolucionales, que permiten el desplazamiento de una ventana sobre la o las matrices obtenidas.
   3.1. Corresponde a las herramientas de que realizan los cálculos necesarios para obtener un objetivo específico, es donde se diseña el modelo con un objetivo concreto (por ejemplo, si deseara detectar señales de tránsito en fotos de carreteras).
4. Corresponde a la salida o lo que se espera obtener. Es decir, es el resultado del modelo al aplicar operaciones sobre la imagen de entrada.´

2.2. Trabajos relacionados

Según P. Sharma, Berwal y Ghai, para 2050 la población mundial alcanzará los 9000 millones de habitantes [10]. Esto supone un reto, será necesario desarrollar métodos que permitan aprovisionar y controlar de forma más eficiente los cultivos, para optimizar la producción y dar abasto a la inmensa población. Una de las necesidades en el control de plantaciones es la mitigación de amenazas, como enfermedades, plagas, etc.

En este escenario, los modelos de Aprendizaje Automático proveen una herramienta para la detección de anomalías a través de la visión por computación y las Redes Neuronales Convolucionales, basándose en Azman e Ismail [3], las *CNN's* han permitido estudiar los niveles de madurez del fruto, esto gracias a la capacidad de estos modelos para el procesamiento visual, concretamente utilizando técnicas de segmentación semántica.

Actualmente existe una amplia variedad de soluciones en el campo del Aprendizaje Automático , para este trabajo nos enfocaremos en el trabajo de *Facebook AI Research*

o **FAIR**, que propone las bases teóricas de Mask R-CNN [6], ya mencionado anteriormente.

Afortunadamente existen implementaciones de *Mask R-CNN*, W. Abdulla realizó un trabajo [2] sobre este tipo de redes neuronales, que aporta herramientas relevantes para la segmentación semántica. Este trabajo aportará una base sólida para la detección de la enfermedad *Fusarium* y *Phytophtora* utilizando tecnología cotidiana como los teléfonos inteligentes.

Capítulo 3

3. Solución planteada

En este capítulo se presenta una visión general de la propuesta, y también se recopilan temas de gestión, análisis de riesgos y ciclo vida del proyecto.

3.1. Propuesta

Se propone el desarrollo de una aplicación móvil que permita analizar fotografías de plantas de piña para realizar la detección de la enfermedad *Fusarium* de forma inmediata. La propuesta se compone de las siguientes partes:

- **Modelo de aprendizaje automático.** Corresponde al módulo encargado del análisis de datos proporcionado. Se desarrolla en Python con marcos de trabajo de aprendizaje automático como TensorFlow [1] o PyTorch [9].
- **Aplicación móvil.** Corresponde a la capa de interacción del usuario final. Hace uso del modelo de aprendizaje automático y brinda funcionalidades adicionales a la detección. Será desarrollada en la plataforma Android con el lenguaje de programación *Kotlin*.
- **Servidor web.** Se refiere a la aplicación de servidor o *backend* que permite la ejecución de tareas auxiliares de la aplicación móvil.

3.2. Involucrados

En el cuadro 3.1 se indican los involucrados en el proyecto.

Nombre	**Departamento**	**Rol**	**Criterio de éxito**
Cortés, Warter	Ingeniería en Computación	Practicante	Desarrollar aplicación web con una alta efectividad en la detección de la enfermedad *Fusarium* .
Gadea, Arnoldo	Escuela de Agronomía	Asesor	Producto de calidad en la detección de la enfermedad *Fusarium*
González, Rogelio	Ingeniería en Computación	Líder del proyecto	Aplicación Android con alta efectividad en la detección de la enfermedad *Fusarium* .
Rivas, Miguel	Ingeniería en Computación	Practicante	Desarrollar aplicación móvil con una alta efectividad en la detección de la enfermedad *Fusarium* .

Cuadro 3.1: Lista de involucrados

3.3. Procedimiento metodológico

En el cuadro 3.2 se describen los indicadores de éxito acordes al alcance del proyecto.

Objetivo específico	**Tarea**	**Meta**	**Indicador**
Recopilar un conjunto de datos de fotografías de piña etiquetadas para alimentar un modelo de aprendizaje automático de detección de enfermedades.	(a) Recopilar fotografías de la enfermedad *Fusarium* . (b) Etiquetar las fotografías recopiladas.	(a) Recopilación de 2000 imágenes. (b) Etiquetado de las imagénes.	(a) Se realiza recopilación en campo. (b) Se cuenta con imágenes etiquetadas.
Diseñar un modelo de aprendizaje automático que permita detectar la enfermedades *Fusarium* y *Phytophtora* en las plantas de piñas a través de fotografías etiquetadas con segmentación semántica.	(a) Diseño de la red neuronal. (b) Entrenamiento de la red neuronal	(a) Prototipo de la red neuronal. (b) La red neuronal detecta la enfermedad *Fusarium* y *Phytophtora* .	(a) Se tiene una arquitectura de red neuronal convolucional. (b) La efectividad del entrenamiento es superior a 80 %.
Desarrollar una aplicación móvil nativa Android para la detección de la enfermedades *Fusarium* y *Phytophtora* en las plantas de piña.	(a) Arquitectura de la aplicación. (b) Desarrollo de la aplicación.	(a) Uso de arquitectura estándar de la plataforma. (b) Aplicación compatible con dispositivos Android.	(a) La arquitectura permite el desacoplamiento de componentes de la aplicación. (b) La aplicación funciona en dispositivos Android.

Cuadro 3.2: Matriz para el procedimiento metodológico

3.4. Análisis de los riesgos

En el cuadro 3.3 se resumen los potenciales riesgos para el proyecto.

Descripción del riesgo	Probabilidad	Riesgo	Estrategia
Modelo de baja eficiencia	Baja	Alto	**Asumir:** aumentar la cantidad de iteraciones de entrenamiento, así como también realizar ajustes menores en el modelo.
Aplicación inestable	Baja	Medio	**Prevenir:** diseñar la aplicación utilizando los estándares de desarrollo de la plataforma.
Inviabilidad de ejecución de modelo en dispositivo móvil	Media	Medio	**Transferir:** El modelo se implementa de lado de la capa de servidor, de esta forma sería necesario acceso a Internet en la aplicación móvil.

Cuadro 3.3: Análisis de riesgos

3.5. Cronograma de trabajo

En la figura 3.1 se describe el flujo de vida del proyecto con base a la ventana de tiempo del II Semestre de 2021.

Name	Duration	Start	Finish
Aplicación móvil Android para la dete	77 days	**7/27/21 8:00 AM**	**11/10/21 5:00 PM**
Elaboración de dataset	27 days	**7/27/21 8:00 AM**	**9/1/21 5:00 PM**
Preparaciones técnicas	7 days	7/27/21 8:00 AM	8/4/21 5:00 PM
Recolección de campo de los datos	5 days	8/5/21 8:00 AM	8/11/21 5:00 PM
Etiquetado de las fotografías	15 days	8/12/21 8:00 AM	9/1/21 5:00 PM
Modelo de aprendizaje automático	18 days	**9/2/21 8:00 AM**	**9/27/21 5:00 PM**
Diseño del modelo	10 days	9/2/21 8:00 AM	9/15/21 5:00 PM
Entrenamiento del modelo	5 days	9/16/21 8:00 AM	9/22/21 5:00 PM
Evaluación del modelo	3 days	9/23/21 8:00 AM	9/27/21 5:00 PM
Aplicación Android	32 days	**9/28/21 8:00 AM**	**11/10/21 5:00 PM**
Definición de requerimientos	5 days	9/28/21 8:00 AM	10/4/21 5:00 PM
Diseño técnico de la aplicación	5 days	10/5/21 8:00 AM	10/11/21 5:00 PM
Desarrollo de la aplicación	20 days	10/12/21 8:00 AM	11/8/21 5:00 PM
Evaluación de la aplicación	2 days	11/9/21 8:00 AM	11/10/21 5:00 PM

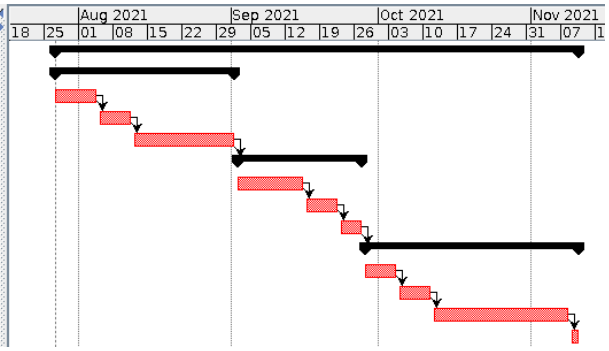

Figura 3.1: Cronograma de trabajo del proyecto.

Capítulo 4

4. Requerimientos y Diseño

En este capítulo se definen los alcances esperados, y el diseño técnico de los componentes del proyecto.

4.1. Definición de requerimientos

En esta sección se describen los requerimientos necesarios para la elaboración del producto final. La tabla 4.1 enumera y detalla dichos requerimientos.

Código	**Nombre**	**Descripción**	**Prioridad**
R-01	Recopilación de fotografías	Se debe recopilar un conjunto de fotografías de 3000 unidades.	Alta
R-02	Carga y etiquetado de fotografías	Se deben cargar las imágenes a una plataforma que permita la etiquetar las fotografías con segmentación semántica	Alta
R-03	Diseño de modelo de Aprendizaje Automático	Se debe diseñar un modelo **Mask R-CNN** que detecte la enfermedad *Fusarium* .	Alta
R-04	Entrenamiento y evaluación del modelo de Aprendizaje Automático	Se debe entrenar y evaluar la eficacia del modelo	Alta
R-05	Exportación del modelo	Se debe exportar el modelo a una para su posterior ejecución en entornos de producción.	Alta

R-06	Creación de servidor	Se debe crear una aplicación de servidor que permita alojar y ejecutar operaciones de inferencia sobre el modelo exportado.	Alta
R-07	Creación de proyecto móvil	Se debe crear una aplicación Android nativa usando el lenguaje de programación *Kotlin*.	Alta
R-09	Vista principal	Se debe crear una vista de inicio que permita visualizar los experimentos realizados previamente, además de una acción que permita iniciar un nuevo experimento	Alta
R-10	Creación de experimentos	El usuario podrá crear experimentos con las siguientes características: ▪ Nombre. ▪ Notas. ▪ Lista de muestras. ▪ Fecha y hora de creación.	Alta
R-11	Creación de muestras	El usuario podrá crear muestras en los experimentos, utilizando la cámara del dispositivo o leyéndolas desde el almacenamiento del dispositivo. Se deben tener al menos los siguientes atributos: ▪ Nombre. ▪ Referencia al dato crudo de la muestra (imagen). ▪ Geolocalización (si está disponible). ▪ Fecha y hora de creación.	Alta
R-12	Historial de experimentos	La aplicación almacenará un historial local de los experimentos realizados a lo largo del tiempo.	Media

R-13	Análisis de experimentos	El usuario podrá ejecutar el análisis de las muestras contenidas en el experimento, a través del servidor de predicciones de Fusapp.	Alta
R-14	Evaluación de la aplicación	Se deben realizar pruebas de rendimiento y estabilidad de la aplicación.	Media

Cuadro 4.1: Requerimientos del proyecto

4.2. Diseño de la plataforma de software

En esta sección se describe la arquitectura que se implementará en las diferentes partes que componen el prototipo.

Modelo de aprendizaje automático

Para el desarrollo de este módulo, es necesario organizarlo en dos etapas: recolección de datos y el diseño e implementación del modelo.

- **Recolección y preparación de datos.** La generación de datos es el punto de partida para el desarrollo de un modelo. Por esto, se recolectan aproximadamente **2000 imágenes** de plantas enfermas, para el etiquetado utilizando segmentación semántica, que se realiza en la herramienta *Labelbox*.
- **Modelo de Aprendizaje Automático.** Se hace uso de una Red Neuronal Convolucional (*CNN* por su acrónimo en inglés) multicapa, esto con el fin de aprovechar los beneficios de este tipo de redes en el análisis y detección de objetos en imágenes. A su vez, se hace uso de la técnica de máscaras de regiones, o *Mask R-CNN*, que permite trabajar sobre segmentos específicos de las imágenes.
  Estas características se adaptan a las necesidades de análisis de objetos del proyecto, el cuál corresponde a la detección de plantas de piña enfermas.

Servidor

El servidor consiste en una aplicación de basada en la nube, enfocada en brindar soporte a tareas auxiliares de la aplicación. Para el ámbito de este proyecto, correspondería al alojamiento de versiones del modelo de Aprendizaje Automático .

Aplicación móvil

Se desarrolla una aplicación nativa para Android, que hace uso de características de aprendizaje automático proporcionadas por el sistema operativo. Se hace uso del lenguaje

de programación Kotlin. También se hace uso de las librerías *lite* de PyTorch para Android.

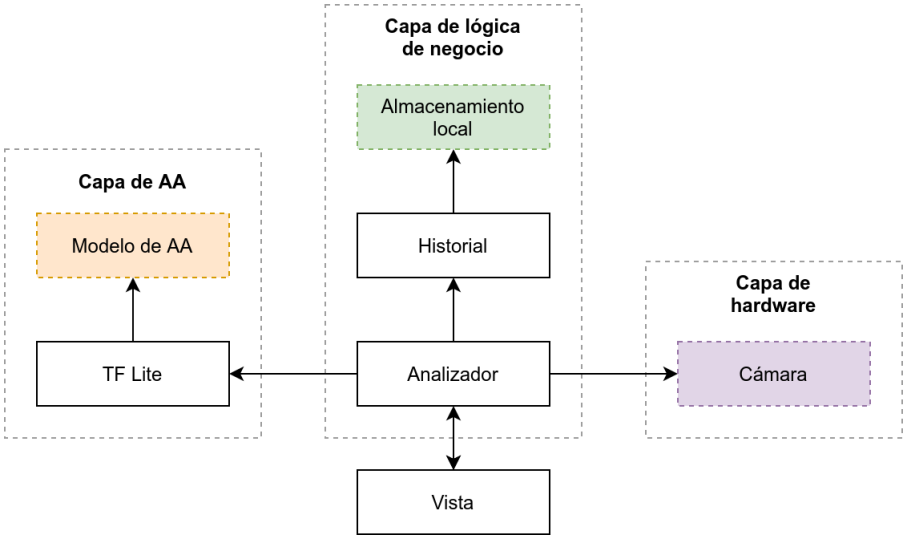

Figura 4.1: Arquitectura de la aplicación móvil.

Diseño extraordinario

Durante la implementación de la propuesta anterior se encontraron algunos problemas relacionados la capacidad operativa del modelo en dispositivos móviles, fue necesario realizar modificaciones en el diseño inicial, cabe resaltar que esta posibilidad ya se había contemplado en el Análisis de riesgos 3.3. A continuación se describen los cambios realizados.

- La aplicación cliente no alojará el modelo de Aprendizaje Automático.
- El modelo se ha trasladado a la capa de servidor y es operable a través de HTTP.
- El servidor ya no incluirá la funcionalidad de almacenar versiones del modelo móvil, sin embargo, no limita su inclusión en futuras versiones.
- La aplicación móvil requiere conexión de red para realizar predicciones.

Capítulo 5

5. Plataforma de software

En este capítulo se registran los resultados de la implementación y evaluación del proyecto.

5.1. Aplicación

Como se describió en el capítulo anterior, este trabajo se compone de tres componentes, el diseño e implementación del modelo de Aprendizaje Automático (incluida la recolección de datos); el desarrollo de una aplicación móvil Android que permite realizar pruebas de campo; y el servidor que aporta funcionalidad esencial para la aplicación móvil. En esta sección se describirá en detalle la elaboración de dichas etapas o componentes, que suman el proyecto denominado **Fusapp**.

Recolección y preparación de datos.

Esta tarea corresponde al punto de partida del proyecto, en esta etapa se llevó a cabo la recolección de fotografías, trabajo se realizó en <**especificar lugar**>, donde se realizaron tomas áreas de fotografía y video utilizando un dron.
Posteriormente, junto a las fotografías tomadas, se extrayeron los fotogramas de los videos capturados, sumando un total de **1871** áreas en resolución 3840x2160 píxeles, es decir, 4K, con un tamaño promedio de **3 MB**. A pesar de que inicialmente se pretendía la recolección de 2000 imágenes, esto se descartó para el alcance de este proyecto, porque la cantidad de datos ya era suficiente.

Una vez obtenidas las imágenes, se optó por cargarlas a un lugar que permitiera el marcado de segmentación semántica, por ello, se optó por *Labelbox*, una plataforma compartida enfocada a la creación y administración de datos de entrenamiento.

Se realizó el etiquetado de 200 imágenes para las enfermedades *Fusarium* y *Phytophtora*
.

Diseño del modelo

Por la naturaleza del proyecto, enfocado en el Aprendizaje Automático , se decidió utilizar *Google Colab*, una herramienta que permite escribir y ejecutar cuadernos de *Jupyter*, aptos para la elaboración de trabajos relacionados a las ciencias de datos.

Se tomó ventaja del API de Labelbox, de esta forma se agilizó el proceso de diseño y entrenamiento del modelo. Concretamente, la aplicación de predicciones fue diseñada sobre el kit PyTorch de Facebook, que proporciona herramientas muy útiles para el diseño de redes neuronales.

A su vez, se hace uso de *Detectron2* [13], un conjunto de funcionalidades que facilitan la creación de modelos basados en la detección de objetos a través de Visión por Computación. *Detectron2* depende de *PyTorch* y *TorchVision* específicamente.

El diseño se ha basado en la red neuronal convolucional ResNet-50 [7], con forma $(1, 3, 224, 224)$, que posteriormente fue entrenada en 1000 épocas.

Entrenamiento

Debido al enfoque en Redes Neuronales Convolucionales y Segmentación Semántica, se hace uso de herramientas de terceros que permitan el etiquetado semántico de los datos, es por ello, que el punto de partida es la plataforma *Labelbox*.

En el cuadro 5.1 se registran los parámetros de ejecución del modelo.

Parámetro	**Valor**
Total de imágenes	200
Total de objetos etiquetados	500
Cantidad de objetos tipo *Fusarium*	340
Cantidad de objetos tipo *Phytophtora*	160
Total de épocas de entrenamiento	1000

Cuadro 5.1: Parámetros de entrenamiento del modelo *Fusapp*

Para efectos de este proceso, se utilizó el kit de entrenamiento por defecto de Detectron2, usando las configuraciones del modelo ResNet-50.

Exportación del modelo

Una vez finalizado el desarrollo del modelo, es necesario realizar la exportación del mismo para la ejecución en entornos de producción.

Para ello, se utilizaron las funcionalides brindadas por PyTorch. A pesar de realizar exportaciones a plataformas móviles, esto dejó de ser necesario debido a la implementación de lado del servidor, por lo que dicha exportación se ha limitado al almacenamiento de los pesos del modelo, refinados por las iteraciones de entrenamiento.

Gracias a la simplificación de exportación del modelo, basta con alojar el archivo de pesos en un repositorio de archivos común, y posteriormente compilar una imagen de Docker capaz de cargar el estado del modelo, lo demás corresponde a tareas de despliegue tradicionales.

El modelo ha sido cargado a un contenedor de archivos de *Google Cloud Storage*.

Servidor

La capa de servidor tiene como objetivo servir la disponibilidad del modelo a través de un servicio web, es decir, mediante un API HTTP.

Esta aplicación ha sido desarrollada en el lenguaje *Python*, en beneficio de usar las librerías necesarias para la operación del modelo. También se usa *Flask* como servidor web, de esta forma es posible realizar solicitudes mediante REST, para obtener resultados de predicción.

En la figura 5.1 se detalla de manera gráfica la arquitectura del servidor.

Despliegue del servidor

Esta herramienta se ha optimizado para el despliegue mediante contenedores en entornos como *Kubernetes*, de forma que se incluye un *script* de construcción de la imagen.

Para efectos de este alcance, el despliegue se realizó mediante Docker en el sistema operativo Ubuntu Server 20.04 basado en el kernel de Linux.

El código fuente del servidor incluye un archivo para crear una imagen de Docker. Asumiento que se encuentra en el directorio raíz del servidor, puede construir la imagen mediante la siguiente instrucción:

```
docker build -t fusapp/server .
```

Donde, `fusapp/server` corresponde al nombre de la imagen de Docker.

Finalmente, para ejecutar esta aplicación, basta con la siguiente instrucción:

```
docker run -d -p {PORT}:3434 fusapp/server
```

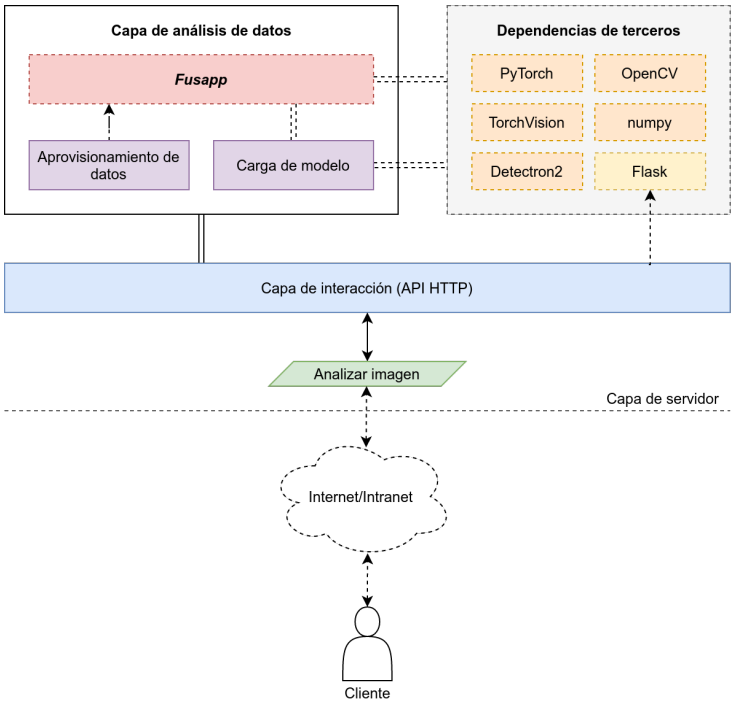

Figura 5.1: Arquitectura de la aplicación móvil.

Aclarando que, `-d` corresponde a una instrucción para ejecutar el contenedor en segundo plano, `{PORT}` es un puerto TCP que el usuario puede decidir, y `fusapp/server` se refiere al nombre de la imagen que compilamos en la instrucción anterior, algunos de estos parámetros pueden ser ajustados a las necesidades que se tengan, asumiendo que se cuentan con los conocimientos generales de administración de aplicaciones contenerizadas.

Por otra parte, también se incluye un *script* `docker-compose.yml` que permite realizar el despliegue completo (equivalente al proceso anterior) con el siguiente comando:

```
docker-compose up --build -d
```

Para conocer en detalle el comportamiento de Docker, se recomienda consultar la referencia del mismo [8].

Aplicación móvil

Fusapp es un producto que tiene como objetivo ser un kit de experimentos para la detección de las enfermedades *Fusarium* y *Phytophtora* a través del uso de fotografías capturadas con el dispositivo, o alojadas en el mismo. Esto se lleva a cabo a través del concepto de experimentos, que representan conjuntos de muestras, correspondientes a las fotografías.

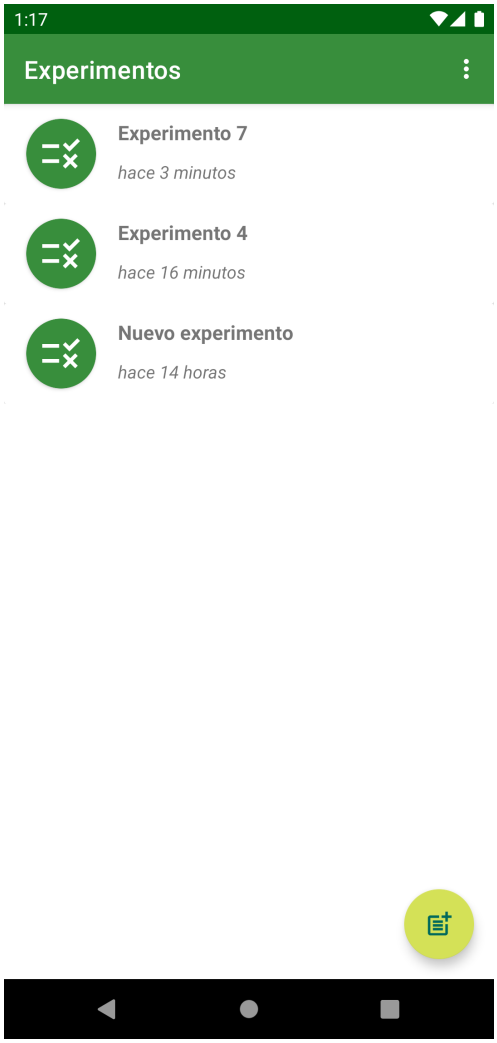

Figura 5.2: Vista principal de la aplicación Fusapp

Fusapp es una aplicación Android nativa, escrita en el lenguaje de programación Kotlin, también depende de librerías de Google y terceros que permiten conseguir la finalidad del proyecto.

A pesar de que en la propuesta inicial, se incluiría el modelo en versión móvil, se ha optado por delegar esta funcionalidad. Esta decisión se tomó debido a los problemas de rendimiento encontrados al momento de ejecutar el modelo en dispositivos móviles.

A nivel de funcionalidad, Fusapp es una herramienta enfocada en la realización de experimentos de detección de las enfermedades ya mencionadas, dichos experimentos contienen una serie de muestras que son capturadas por el usuario. Después de la captura de datos, se procede al análisis, proceso que requiere la comunicación con el servidor.

Este diseño fundamentado en el concepto de experimento, permite plasmar de cara al usuario, una interfaz gráfica simple, que consigue brindar al usuario las siguientes capacidades esenciales:

- Crear experimentos que permiten alojar múltiples muestras.
- Realizar análisis de las muestras contenidas en los experimentos.
- Acceder a un historial de experimentos realizados.

En la figura 5.2 se muestra la vista principal de la aplicación Fusapp.

La figura 5.3 muestra el flujo de acceso a los datos a través de la arquitectura de inyección de dependencias utilizada.

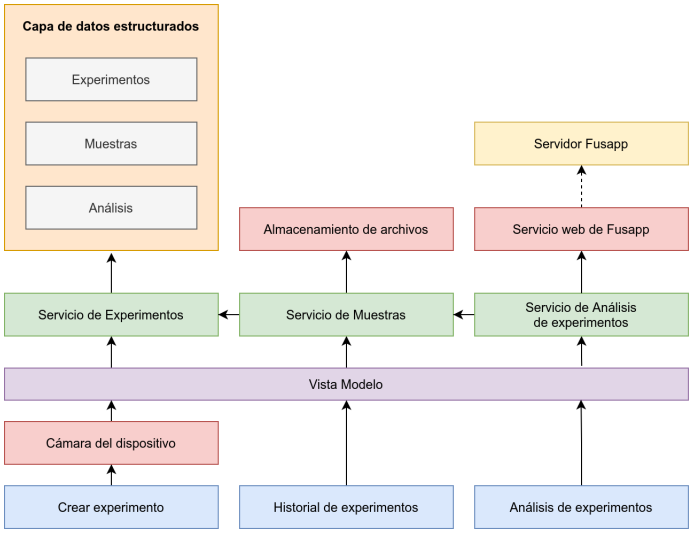

Figura 5.3: Arquitectura de componentes de la aplicación móvil.

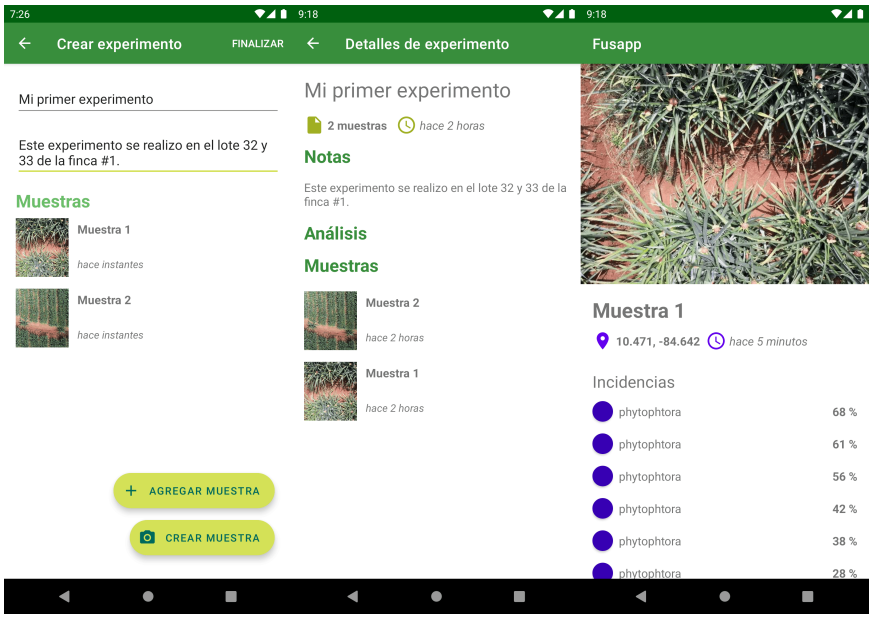

Figura 5.4: Algunas vistas de la aplicación.

En la figura 5.4, de izquierda a derecha se muestra la vista de creación de experimentos, el detalle del experimento creado, y finalmente los detalles de análisis de una muestra.

5.2. Evaluación

En este apartado se describen los resultados de la implementación del proyecto, para ello se analizarán algunas métricas generadas por el producto creado.

Modelo de Aprendizaje Automático

A pesar de las dificultades durante el diseño e implementación del modelo, ha sido posible obtener unos resultados modestos para este prototipo.

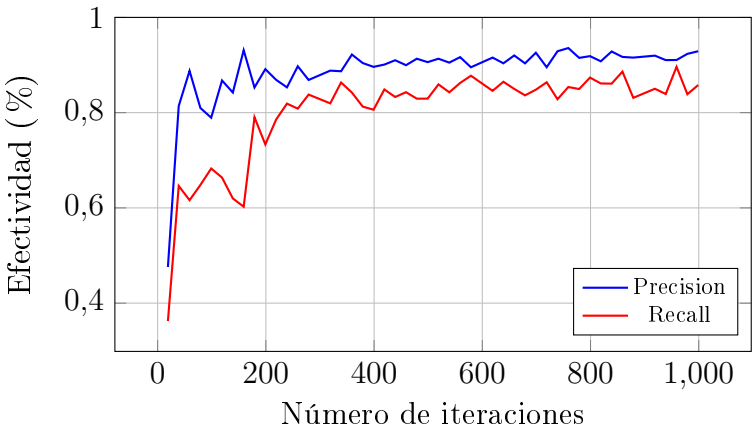

Figura 5.5: Métricas de Precisión, Exhaustividad *(Recall)* del modelo.

En la figura 5.5 se observan las métricas de **precisión** y **exhaustividad**, estos parámetros nos indican la calidad de la predicción y la capacidad de identificación del modelo respectivamente.

Adicionalmente, en la figura 5.6 se muestra la métrica F1, que corresponde a un promedio entre las métricas de precisión y exhaustividad mencionadas anteriormente.

Es relevante observar la estabilidad continuada a lo largo del entrenamiento, manteniendo los números por encima del 90 %, en total se realizaron 1000 iteraciones de entrenamiento.

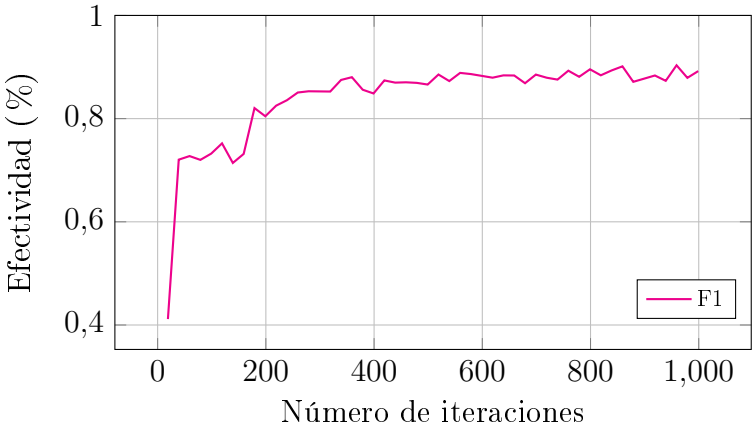

Figura 5.6: Métrica de F1.

Servidor

La evaluación del servidor se realizó mediante un despliegue de Docker en una máquina virtual de Google Cloud.
A diferencia de la etapa de entrenamiento, en esta implementación, el modelo se ejecuta sobre CPU, con el fin de ahorrar en costos de infraestructura.

En el cuadro 5.2 se resume la configuración de la máquina virtual.

Parámetro	**Valor**
Sistema operativo	Ubuntu Server 20.04
CPU	2 × Intel Xeon (R) @ 2.20 GHz, caché 56 MB
GPU	N/A
RAM	4 GB
Almacenamiento	32 GB @ 192 IOPS lectura/escritura

Cuadro 5.2: Especificaciones de máquina virtual utilizada para desplegar servidor.

Rendimiento de modelo en aplicación móvil

Al momento de desplegar el modelo en la app Android, se encontraron una serie de problemas relacionados al rendimiento del dispositivo y la ejecución de predicciones. Se identificaron problemas de memoria en la depuración, que en general mostraban un mensaje similar al siguiente:

```
E/AndroidRuntime: FATAL EXCEPTION: main
Process: cr.ac.tec.fusapp, PID: 22407
java.lang.OutOfMemoryError: Failed to allocate a
383533072 byte allocation with 25165824 free bytes
and 141MB until OOM, target footprint 413191376, growth
limit 536870912
```

Se procedió a utilizar las herramientas de perfilado de rendimiento de Android Studio, y se conocieron algunas métricas de consumo de CPU y memoria que permitieron entender lo que ocurría.

En las figuras 5.7, 5.8 y 5.9 el primer punto rojo denota la carga del modelo, mientras que el segundo corresponde a la ejecución de inferencia sobre una imagen. Se puede observar que el consumo de recursos se dispara a porcentajes inviables durante un 90 segundos de pruebas.

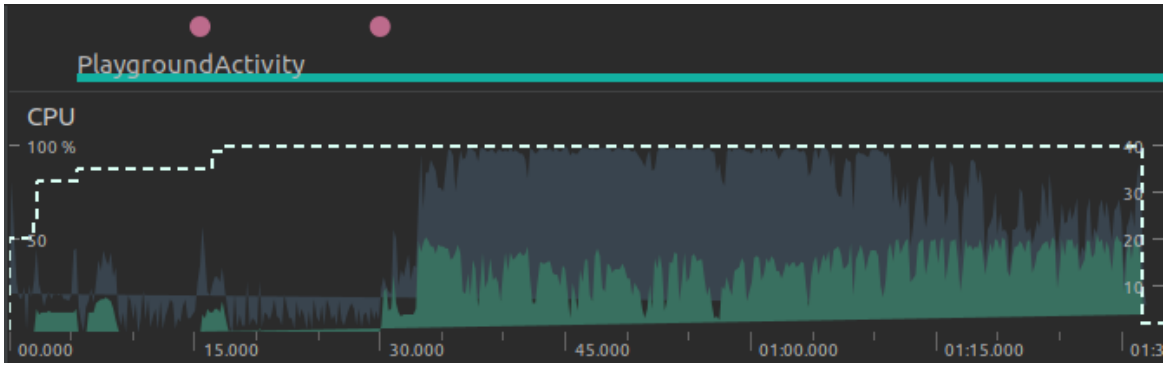

Figura 5.7: Consumo de CPU de Fusapp durante operaciones relacionadas al modelo móvil.

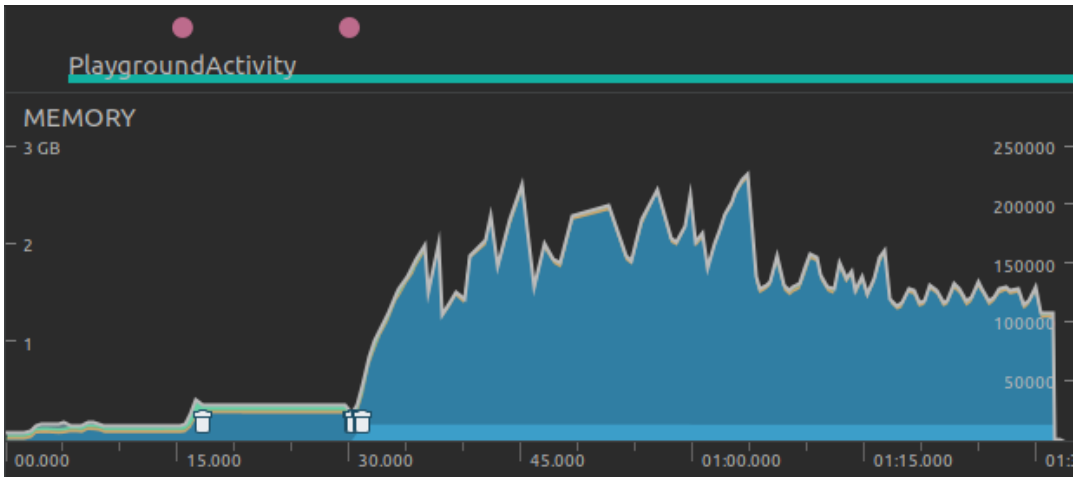

Figura 5.8: Consumo de memoria de Fusapp durante operaciones relacionadas al modelo móvil.

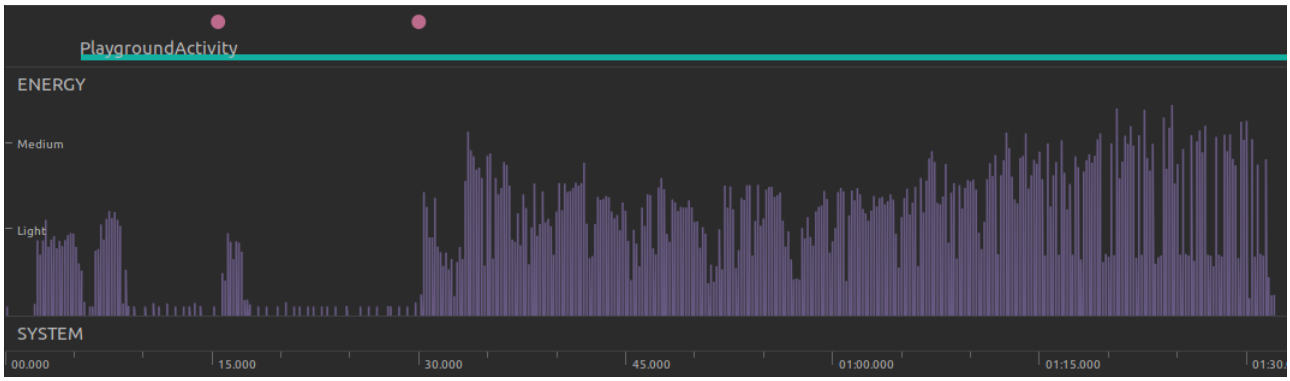

Figura 5.9: Consumo de energía de Fusapp durante operaciones relacionadas al modelo móvil.

Capítulo 6

6. Conclusiones

En este capítulo se documenta algunos aprendizajes, apuntes y recomendaciones relacionadas al proyecto.

6.1. Recomendaciones

En este apartado, abordaremos algunas recomendaciones sobre el estado actual del proyecto, así como ideas que podrían dar continuidad para el futuro desarrollo de un producto, además de algunos apuntes y notas técnicas.

Apuntes

- La combinación de las disciplinas investigativas y de software comercial de consumo, permite diseñar prototipos y entregar soluciones innovadoras. Por ello es importante la estimulación de estos espacios que permiten el desarrollo de estos conocimientos, tanto de caracter industrial como investigativo.
- El dominio matemático tiene importante relevancia al momento de trabajar en áreas relacionadas a las Ciencias de datos o Inteligencia Artificial. Los conocimientos de áreas como el álgebra lineal o la estadística permiten comprender el trasfondo de los modelos diseñadas.

Notas técnicas

Esta es una serie de consideraciones técnicas que refieren al desarrollo del proyecto.

- El uso de plataformas compartidas y de nube permite mayor flexibilidad al momento de gestionar el proyecto, esto incluye herramientas de colaboración compartida como Labelbox para etiquetar datos, o Google Colab, para la creación de cuadernos de datos científicos. También es relevante mencionar que los servicios de nube, es decir, infraestructura como servicio, permiten elaborar entornos de pruebas mediante servicios como el almacenamiento de archivos o máquinas virtuales, sin embargo, estos

tienen costos monetarios, que podrían ser pequeños si se hace un uso moderado del servicio.

- La contenerización de aplicaciones es una solución fácil y potente que permite realizar despliegues de aplicaciones de forma rápida.

Continuidad del proyecto

A pesar de que inicialmente se pretendía el desarrollo de una aplicación capaz de ejecutar predicciones de segmentación semántica sin depender de infraestructura externa, el resultado fue diferente. La implementación de un servidor de predicciones permitió observar otros escenarios de interés.

A pesar de no estar presente en ninguna funcionalidad del estado actual de la aplicación, se crearon estructuras relacionadas a datos geográficos. La intención de esto es la posibilidad de crear experimentos de campo asociados a espacios geográficos, que a su vez también incluyen datos de fecha y los resultados experimentales.

Sin embargo, este enfoque reduce la necesidad de ejecutar el modelo en el dispositivo móvil, debido a que se preveé que sería ideal utilizar una capa de servidor que permita gestionar dichos experimentos.

Bibliografía

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Waleed Abdulla. Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. `https://github.com/matterport/Mask_RCNN`, 2017.

[3] Ahmad Aizuddin Azman and Fatimah Sham Ismail. Convolutional neural network for optimal pineapple harvesting. ELEKTRIKA- Journal of Electrical Engineering, 16(2):1–4, Aug. 2017.

[4] DEL Cooke, A Drenth, JM Duncan, G Wagels, and CM Brasier. A molecular phylogeny of phytophthora and related oomycetes. Fungal genetics and biology, 30(1):17–32, 2000.

[5] Cámara Nacional de Productores y Exportadores de Piña. Estadísticas | canapep. `https://canapep.com/estadisticas`.

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. CoRR, abs/1703.06870, 2017.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. CoRR, abs/1512.03385, 2015.

[8] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. Linux journal, 2014(239):2, 2014.

[9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani,

Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 8024–8035. Curran Associates, Inc., 2019.

[10] Parul Sharma, Yash Paul Singh Berwal, and Wiqas Ghai. Performance analysis of deep learning cnn models for disease detection in plants using image segmentation. Information Processing in Agriculture, 7(4):566–574, 2020.

[11] Koczyk G. & Waśkiewicz A. Stępień, Ł. Diversity of fusarium species and mycotoxins contaminating pineapple. J Appl Genetics 54, pages 367–380.

[12] Cecilia Tapia and José Amaro. Género Fusarium. Revista chilena de infectología, 31:85 – 86, 02 2014.

[13] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

Anexos

Glosario

- ***API.*** Corresponde a Interfaz de Programación de Aplicaciones, por su acrónimo en inglés de *Application Programming Interface*. Se refiere a la capa de interacción de un servicio, aplicación o componente. A pesar de ser un término genérico, en la actualidad su uso se puede referir ampliamente a servicios web que usan el protocolo HTTP.
- ***backend.*** Se refiere a la capa de componentes y aplicaciones que hacen de infraestructura y lógica de negocio de un proyecto. El usuario no interactúa directamente con esta capa.
- **contenedor.** Aplicación que se ejecuta en un entorno aislado.
- ***frontend.*** Corresponde a la capa de interacción con el usuario, es decir, la vista. Usualmente interactúa con un *backend* para llevar a cabo las tareas.
- **HTTP.** *Hypertext Transfer Protocol* o Protocolo de Transferencia de Hipertexto es un estándar de transferencia de información sobre el cuál se basa la infraestructura de comunicación moderna.
- ***Kubernetes.*** Sistema de orquestación de contenedores.
- ***script.*** Comúnmente se refiere a una lista de instrucciones para realizar tareas generales.
- **tensor.** Objeto matemático capas de abstraer múltiples dimensiones.