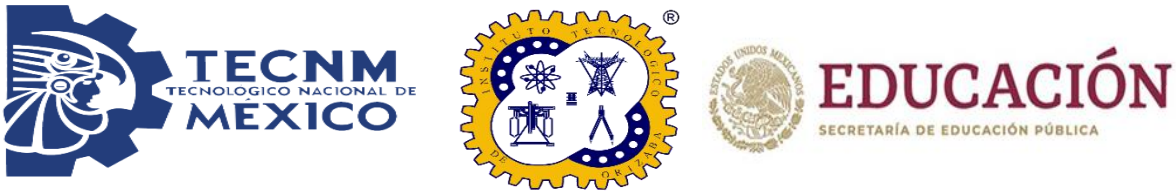


Instituto Tecnológico de Orizaba

División de Estudios de Posgrado e Investigación

Maestría en Sistemas Computacionales

TESIS

"Generación automática de interfaces de usuario multi-dispositivo para el ámbito de e-Commerce y Social Media mediante técnicas de Deep Learning"

PRESENTA:

I.T.C. Aarón Colmenares Morales

PARA OBTENER EL GRADO DE:

Maestro en Sistemas Computacionales

DIRECTOR:

Dr. Giner Alor Hernández

CO-DIRECTOR:

M.S.C. Laura Nely Sánchez Morales

Contenido

Índice de tablas

Índice de Figuras

Índice de Código

Resumen

El aprendizaje profundo incrementa progresivamente su uso día con día, tanto en las computadoras y dispositivos móviles para automatizar las aplicaciones. Gracias al aprendizaje profundo que permite el apoyo en varias áreas como transporte o salud, se incrementó considerablemente el uso de las redes neurales que implica aprendizaje de percepción de objetos, traducción y reconocimiento. El aprendizaje profundo genera una gran ayuda en los ámbitos médicos, traducción de textos o hasta la identificación de caminos, ya que se usan inclusive para coches autónomos. Algo a tomar en cuenta del aprendizaje profundo es la precisión al momento de identificar objetos, por la tanto, la precisión satisface la necesidad de los usuarios en el ámbito de consumo electrónico.

El objetivo del presente trabajo de investigación es desarrollar un módulo para la generación automática de interfaces de usuario las cuales se desplieguen en diferentes dispositivos, esto es, a partir de una imagen digital identificar mediante técnicas de aprendizaje profundo los elementos en la imagen para aplicaciones de dominios *e-Commerce* y *Social media*.

Como principal beneficio que conlleva el trabajo al desarrollar el módulo, es tener aplicaciones dinámicas que permitan integrarse en el dominio *e-Commerce* y *Social media*. La aplicación del aprendizaje profundo como parte de la inteligencia artificial corresponde a una transformación digital, que permite ser más eficiente para el usuario final y el desarrollador de software, además de reducir errores humanos y costo de desarrollo.

Abstract

Deep learning is gradually increasing the use day by day on computers and mobile device to automate applications. Thanks to deep learning that significantly increased the use of the neuronal networks to compete to learn object perception, translate an recognize. Deep learning gives a huge help on medical checks, translations of text or even identification of roads for autonomous cars. To take deep learning into account is the precision when identifying, therefore, the precision satisfies the needs of the users on improving the performance of electrical consumption.

The goal for this paper propones is a develop module for automatic generation of user interfaces which are deployed on different devices from a digital image for e-Commerce and Social Media domains through deep learning techniques.

The main benefit of develop the module, it is working a dynamic application that can be fully integrated into e-Commerce y Social Media domains. The application of deep learning as part of artificial intelligence corresponds to the digital transformation because it allows be more efficient to the end user and the software developer in addition to reducing human errors and development cost.

Introducción

Día a día se generan nuevos y variados métodos en el desarrollo de software más rápidos, certeros y confiables. Desde el punto de vista de la ingeniería del software y de forma específica dentro de la etapa de diseño de software, los UIDPs son una solución recurrente que resuelve problemas comunes de diseño, como son: las resoluciones de diferentes dispositivos, el diseño responsivo o problemas de navegación; todas estas soluciones se plantean a través de un lenguaje común para los diseñadores. Por su parte, el uso de aplicaciones de dominios como *e-Commerce* y *Social medi,* cuentan con una gran demanda para los usuarios, debido a que se exapenden los nichos de mercado cada vez más al entorno social.

Por otra parte, el *Deep Learning* es un subconjunto de técnicas de inteligencia artificial que permite diseñar modelos computacionales compuestos de múltiples capas para aprender un conjunto de datos etiquetados, con el contenido de varias capas permiten el reconocimiento del lenguaje, la detección de algún objeto e inclusive la traducción. Dentro del *Deep Learning* encontramos las CNN (*Convolutional Neural Networks*, Red Neuronal Convolucional), las cuales son redes neuronales para procesar datos que llegan en forma de múltiples arreglos, y generar una rápida detección de bordes, líneas, entre otras características simples.

De acuerdo a la literatura revisada en este documento existen generadores de software similares y se encuentra orientados a otros dominios, existen también procesos poco orientados para las aplicaciones *e-Commerce* y *Social Media* que manipulen técnicas de *Deep Learning* y ayuden en el reconocimiento de UIDPs en imágenes. Desde esta perspectiva, la identificación de cada UIDP es importante para clasificar y determinar el tipo de dispositivo adecuado para su funcionamiento. Tal situación, motiva a plantear un proceso de generación automática de aplicaciones basado en técnicas de *Deep Learning*.

El primer pilar de tres en esta investigación, es generar software y aplicar el uso de una imagen o un boceto generado a mano alzada. El segundo pilar, es el desarrollo de software basado en el aprendizaje profundo, donde una de las características principales que se aprovechó en el desarrollo de software es el aprendizaje a través del entrenamiento, el cual mejora el tiempo de desarrollo. El tercer pilar son los UIDPs, ya que permiten identificar y restaurar ciertos errores de una aplicación, tales como mantener una resolución estándar y

rectificar errores generados de diseño por el desarrollador, cumpliendo con una aplicación fácil de usar para el usuario.

Como contribuciones para la ingeniería de software, se tienen cuatro aspectos importantes: (1) generar beneficios para que el proceso de desarrollo de software sea mucho más intuitivo y que sea tan bueno como lo hace un ser humano, (2) diseñar un modelo de *machine learning* y un conjunto de reglas que permitan la generación automática de código, (3) tener un mejor desempeño para aumentar la precisión en la clasificación de imágenes que mejore los ciclos de desarrollo del software y (4) enriquecer las aplicaciones que se sumen a las actividades diarias y aumente la motivación para el desarrollo de software.

Este documento está estructurado de la siguiente manera, en el capítulo 1 se detallan los antecedentes y su marco conceptual, en el capítulo 2 se presenta el estado del arte referente a los diversos trabajos relacionados para la generación automática de software, divido en dos partes: herramientas para la generación de aplicaciones o desarrollo de software y técnicas *Deep Learrning* en el desarrollo de software. En el capítulo 3 la propuesta de solución, que incluye una descripción de los UIDPs identificados, analizados y aplicados para la generación automática de aplicaciones del dominio *e-Commerce* y *Social Media*. Se describe lo que implica el proceso de generación de software. En el capítulo 4 se definen dos casos de estudio que hacen uso del generador de aplicaciones y la utilización de loa red neuronal utilizada. Finalmente, el capítulo 5 se presentan las conclusiones y recomendaciones del trabajao de investigación.

Capítulo 1 Antecedentes

1.1. Marco Conceptual

En la siguiente sección se abordan los términos destacados y relacionados con el trabajo de investigación. Cada término tiene una breve descripción que permita comprender mejor.

1.1.1. Inteligencia Artificial

La A.I. (*Artificial Intelligence*, Inteligencia Artificial) en términos amplios es la forma de cómo las máquinas desempeñan tareas humanas, tales como aprender, razonar y resolver problemas. Por tal motivo, son sofisticados algoritmos que reconocen patrones en registro de datos no estructurados como las imágenes, el texto y el lenguaje [1].

1.1.1.1. Aprendizaje Automático

Machine Learning o Aprendizaje Automático es derivado como subconjunto de la A.I. [2]. El aprendizaje automático ofrece técnicas y modelos que se seleccionan en función de la aplicación, dependiendo del tamaño de datos para procesar y el tipo de problema a resolver.

1.1.1.2. Aprendizaje Profundo

Derivado como un subconjunto de la A.I. y del aprendizaje automático que utiliza redes neuronales artificiales, el aprendizaje profundo soporta modelos computacionales compuestos de múltiples capas procesadas para aprender representaciones de datos con contenido de varias capas, que permiten el reconocimiento del lenguaje, la detección de algún objeto e inclusive la traducción. El aprendizaje profundo descubre una estructura intricada sobre un conjunto de datos, el usar el algoritmo de retro propagación para indicar cómo la máquina cambia sus parámetros internos para la representación en cada capa desde la representación en una capa previa. Los métodos mantienen un mejoramiento en el reconocimiento del lenguaje, reconocimiento de objetos visuales, detección de objetos y muchos otros dominios tales como descubrimiento de medicamentos y genómicas [3].

Gracias a las grandes cantidades de datos etiquetados y la potencia de cálculo de las GPU (*Graphics Processing Unit*, Unidad de Procesamiento Gráfico) se logra una eficiencia para

cubrir el aprendizaje profundo. Algunas técnicas de aprendizaje profundo actualmente aplicadas son:

- **Conducción autónoma:** detección de objetos y personas, contribuye a reducir accidentes.
- **Sector aeroespacial y de defensa:** identificar objetos desde los satélites en ciertas áreas seguras o no seguras.
- **Investigación médica:** detección de células cancerígenas.
- **Electrónica (CES):** audición automatizada y la traducción del habla.

1.1.1.2.1. Redes Neuronales Convolucionales

Las ConvNet o CNN [3] (*Convolucional Neural Networks*, Redes Neuronales Convolucionales) son diseños para procesar datos que llegan en forma de múltiples arreglos, por ejemplo, una imagen a color compuesta de tres arreglos 2D que contiene intensidades de pixeles en los tres canales de color. Las modalidades de los datos están en la forma de múltiples arreglos: 1D para señales y secuencias para agregar el lenguaje; 2D para imágenes o espectrogramas de audio y 3D para video o imágenes volumétricas. Se mantiene cuatro ideas claves detrás de los ConvNets que toman ventaja de las propiedades de señales locales: conexiones locales, pesos compartidos, agrupación y el uso de muchas capas.

1.1.1.2.1.1. Inception_V3

Inception V3 creada por Google es la tercera versión en una serie de arquitecturas para redes neuronales convolucionales. La red neuronal esta entrenada en usar una colección de 1000 clases que originalmente contiene un millón de instancias para entrenar, la versión de TensorFlow contiene 1001 clases, la cual contiene una clase adicional que no incluye el entrenamiento por ImageNet. Derivado a su gran entrenamiento Inception V3 mantiene el conocimiento de aprendizaje incluso para aprender nuevas clases y aplicar en nuevos pequeños conjuntos de datos, como resultado mantiene una alta clasificación sin necesidad de un extenso entrenamiento y poder computacional.

1.1.1.2.1.2. AlexNet

AlexNet es el nombre de una red convolucional, diseñada por Alex Krizhevsky. La red logra el top-5 de error de 15.3%, más de 10.8 de porcentaje en puntaje menor que los demás. Además, es factible la utilización de unidades de procesamiento gráfico durante el entrenamiento.

1.1.2. *User Interface Design Pattern,* Patrones de Diseño de Interfaz de Usuario (UIDPs)

Los UIDPs son una solución recurrente que resuelve problemas comunes de diseño, como son: las resoluciones de diferentes dispositivos, el diseño responsivo o problemas de navegación; todas estas soluciones se plantean a través de un lenguaje común para los diseñadores. [4].

1.1.2.1. Composición de Interfaces de Usuarios

La composición de interfaces de usuario es el conjunto de interfaces de usuario que permiten englobar y enlazar los UIDPs que involucra una rápida respuesta para la creación final de una aplicación sobre un proyecto en general.

1.1.3. Aplicaciones de Social Media

Los *Social Media* o Medios Sociales permiten la creación e intercambio de contenido generado por el usuario y el diseño de una gama de aplicaciones basadas en Internet [5]. Ejemplos: Facebook®, Twitter®, entre otras.

1.1.3.1.Tipos de aplicación

Para el dominio de Social Media se especifican algunos tipos de aplicaciones como:

- Redes Sociales: se basan en la forma de compartir la información y, además permiten compartir imágenes, videos, entre otras actividades multimedia, un ejemplo de ello es Facebook®.

- Microblogging: se basa en enviar y publicar información reducida un ejemplo claro es Twitter®.
- Streaming: basadas en el envío de información de audio o video en fragmentos y este se reproduce al momento, un ejemplo claro es Spotify®.

1.1.4. Aplicaciones de Comercio Electrónico

Las aplicaciones de *e-Commerce* (*Eletronic Commerce*, Comercio Electrónico) son soluciones de comercio electrónico o comercio por Internet, las cuales son seguras y permiten satisfacer las necesidades de los clientes y empresas. Permiten interactuar con los clientes a través de productos y ofertas personalizados, procesar información de forma rápida y segura. Para enfocarse en el cumplimiento y en dar servicio al cliente se dan soluciones por medio de redes sociales o páginas Web [6].

1.1.4.1. Tipos de aplicación

Para el dominio de *e-Commerce* se especifican algunos tipos de aplicaciones como:

- B2C (*Bussiness to Consumer*, Negocio a Consumidor): donde las aplicaciones se basan en servicios y productos para el cliente final, un ejemplo claro son la venta de los productos por medio de una página de internet como Dell® o Asus®.
- C2C (*Consumer to Consumer*, Consumidor a Consumidor): se basa en aplicaciones para dirigidas para que un cliente final se vincule con otro cliente final, algunos ejemplos específicos son Amazon® o Ebay®.

1.1.6. APIs de aprendizaje profundo

A continuación, se explican algunas APIs (*Application Programming Interface*, interfaz de programación de aplicaciones) que soportan algoritmos de aprendizaje profundo, el uso de bibliotecas o lenguajes de programación que se utilizan para las diferentes herramientas de desarrollo.

1.1.6.1. TensorFlow

Es una biblioteca de código abierto para el cálculo numérico y utilizar gráficos de flujo de datos. Desarrollado por Google en la organización de Inteligencia Artificial para el uso del aprendizaje automático y la investigación de redes neuronales profundas, pero se usa de forma general permitiendo una gran variedad de dominios [7].

TensorFlow es multiplataforma por lo que permite su aplicación en diferentes sistemas GPU (*Graphics Processing Unit*, Unidad de Procesamiento gráfico) y CPUs (*Central Processing Unit*, Unidad Central de Procesamiento). Distribuye la ejecución de motores abstractos que soporta muchos dispositivos y provee un alto rendimiento en el núcleo del lenguaje en C++. Además, es compatible con el lenguaje Python. Las capas de la API proveen una interfaz simple para el uso cómodo de modelos de aprendizaje profundo.

TensorFlow es capaz de construir un gráfico computacional. El gráfico es una estructura de datos que describe completamente el cálculo que se desea realizar, permite muchas ventajas como: portabilidad de ejecutar en cualquier momento, es optimizable para diferentes versiones y soporte para la distribución ejecutable.

Algunos modelos son:

- ***The Object Detection API***: modelo con un núcleo de aprendizaje automático con el reto de crear modelos de aprendizaje automático compatibles para localizar e identificar múltiples objetos una sola imagen.
- ***Tf-seq2seq***: un *framework* de Google® que anunció *Google Neural Machine Translatio*n, un modelo que permite la traducción de secuencia por secuencia y lograr resultados de última generación.
- ***ParseySaurus***: son un conjunto de pre-entrenamientos, los cuales son nuevos modelos de uso de base de caracteres que se ingresan y predicen el resultado de nuevas palabras a base de deletreo usados en contexto.
- ***Multistyle Pastiche Generator***: este modelo amplía la transferencia de imagen al crear una red única que realiza más de una imagen al mismo tiempo.

1.1.6.2. Keras

Keras es una API para redes neuronales de alto nivel, escrito en Python y optimizado para correr con Tensorflow, CNTK o *Theano*, desarrollada con un control para establecer una rápida experimentación. Keras permite conservar el menor tiempo posible para un resultado sin retraso que es la clave para una buena investigación. Algunas características de *Keras* son las siguientes:

- Optimizado para CPU y GPU
- Permite un prototipo fácil y rápido, ya que es amigable, modulable y es extensible
- Soporta redes convolucionales y redes recurrentes, incluso combinaciones.

En la tabla 1.1 se analizan algunas APIs identificadas por plataforma, lenguaje, tipo de biblioteca de GPU, código abierto y si maneja redes convolucionales.

Tabla 1.1 Características de las APIs

API	Plataforma	Lenguaje	Biblioteca GPU	Código Abierto	Desarrollado por	Compañías que la usan	Redes Convolucionales
TensorFlow	Windows® Linux® macOS® iOS® Android®	Python C++ Java	CUDA	Sí	Google®	Google® Ebay® Intel® DropBox® SAP® Uber® Twitter® IBM® NVIDIA®	Sí
Keras	Windows® Linux® macOS®	Python	CUDA, OPENCL	Sí	Open-ended Neuro-Electronic Intelligent Robot Operating System	Google ®	Sí

1.1.7. Lenguaje de programación con soporte para *Deep Learning*

A continuación, se explica el lenguaje de programación principal que soporta el desarrollo de las técnicas *Deep Learning*.

1.1.7.1. Python

Python es un intérprete, orientado a objetos y un lenguaje de programación de alto nivel con semántica dinámica. Construido con una estructura de alto nivel, en combinar un tipado dinámico y un enlace dinámico, crear un atractivo desarrollo rápido de aplicaciones. Se usa comúnmente un *script* o conexión de diálogo para conectar componentes ya existentes. Python es simple, su sintaxis es fácil de aprender, enfatiza legibilidad y por lo tanto reduce el costo de mantenimiento. Soporta además módulos y paquetes, con modularidad de programas y reutilización de código. Contiene disponibilidad de una gran cantidad de bibliotecas y su código es abierto para su libre distribución [11]. Contiene extensiones como: py, pyc, pyd, pyo y pyw.

1.1.8. IDEs

Los IDEs (*Integrated Development Environment*, Entorno de Desarrollo Integrado) son editores de código fuente y proporcionan un servicio integral para facilitar el desarrollo de software al desarrollador o programador. A continuación, se explica el IDE principal que soporta el desarrollo del proyecto de investigación aquí descrito.

1.1.7.1. Spyder

Spyder es un IDE de código abierto para programación científica en el lenguaje Python. Integra un número de paquetes prominentes del repositorio de Python, incluye NumpPy, Scipy, Matplotlib, pandas, IPython, entre otros; contiene la licencia MIT. Pierre RayBaut creó Spyder en 2009, desde el 2012 es mantenido por la comunidad de Python y por desarrolladores científicos. Está disponible para la plataforma Anaconda, a través de Windows ®, macOS ® y Linux ®. Contiene además controladores de Primeros y Terceros con

soporte de inspección de datos y una cantidad de instrumentos introspectivos tales como Pyflakes y Rope

1.1.9. Servidor de aplicaciones Web

A continuación, se explica el servidor de aplicaciones Web principal.

1.1.9.1. Apache HTTPS Server

Apache es el software para servidor Web más utilizado, desarrollado y con soporte por *Apache Software Foundation*, creado dentro del proyecto HTTP Server (httpd). Apache es un software de código abierto disponible de forma gratuita. Funciona en el 67% de todos los servidores web del mundo. Es rápido, confiable y seguro, es altamente personalizado para satisfacer las necesidades de muchos entornos diferentes mediante el uso de extensiones y módulos. La mayoría de los proveedores de alojamiento de WordPress usan Apache como su software de servidor web [18].

1.1.10. Metodología para el desarrollo de software

A continuación, se explica la metodología para el desarrollo del proyecto.

1.1.9.1. Prototipado

Esta metodología de desarrollo de software que permite obtener una versión preliminar del software con el fin de demostración o evaluación. Desde esta perspectiva, para los clientes y usuarios, el obtener una implementación parcial del sistema da la posibilidad de experimentar y retroalimentar con su opinión las ventajas y desventajas del sistema, se realiza una documentación para una posterior revisión y se pone en marcha el desarrollo de una nueva versión que se acerca más a la versión final. De tal manera que un prototipo desarrollado de forma rápida para su ejecución [21]. Algunas características de esta metodología son las siguientes.

- Método informal para el desarrollo.
- El Prototipo alcanza el producto final.

Las fases del prototipado son las siguientes:

- Investigación preliminar: Define los problemas, cuenta con el análisis y especificaciones, diseño y construcción, evaluación y modificación.
- Diseño técnico: Documentación y diseño detallado.
- Programación y prueba. Se encarga de ver las especificaciones del diseño de prueba e implementación.
- Operación y mantenimiento: Etapa de instalación y modificaciones posteriores

1.2. Planteamiento del Problema

El uso continuo y creciente de las *e-Commerce* tales como Amazon®, ebay®, Blue Nile®, Costco®, Mercado Libre® y las *Social media* como Facebook®, Twitter®, Linkedin®, Google+®, Instagram®, Flickr™ YouTube®, entre otras, se demuestra que los usuarios se adaptaron al uso de estos tipos de aplicaciones, por lo que este incremento de usuarios, busca nuevas fuentes para comunicarse o nuevas formas de compartir información. Por otra parte, en la ingeniería de software, de manera específica en el desarrollo de software, la búsqueda de la reducción del tiempo de desarrollo y además el costo que este involucre, es un reto cada vez mayor, se mejora cuando se involucran nuevas técnicas y tecnologías que van más allá del desarrollo tecnológico. De modo que, en esta propuesta se plantea generar de forma automática código fuente de interfaces de usuarios multidispositivo, a partir de una imagen digital que represente una interfaz de usuario (previamente delimitada), para lo cual se identifican composiciones de UIDPs a través de técnicas de aprendizaje profundo. En este sentido, el aprendizaje profundo permite un conjunto de modelos computacionales que imitan el funcionamiento del sistema nervioso humano, los modelos computacionales están compuestos de múltiples capas de procesamiento para aprender representaciones de datos con múltiples niveles de abstracción. El uso de métodos de aprendizaje profundo tiene diversas contribuciones en áreas de procesamiento de imágenes, reconocimiento de voz, audio, video y de manera general en sistemas de aprendizaje automático. Por otra parte, el uso de UIDPs para el desarrollo de interfaces de usuario garantiza una solución a los principales problemas de diseño como las restricciones del tamaño de pantalla para desplegar

el contenido o una navegación adecuada, además de garantizar una adecuada interacción con el usuario. En este mismo sentido, la composición de patrones se refiere a un patrón de diseño que se combine con otro o en su defecto se acompañe de uno o más patrones de diseño.

1.3. Objetivo General

Desarrollar un componente de software que permita la generación automática de interfaces de usuario a través de imágenes con composición de UIDPs y técnicas de aprendizaje profundo.

1.3.1. Objetivos Específicos

1. Estudiar y analizar los diferentes algoritmos de *Deep Learnin*g reportados en la literatura.
2. Analizar e identificar los UIDPs válidos para la composición en interfaces de aplicaciones de *e-Commerce* y *Social media*.
3. Seleccionar los algoritmos adecuados para el reconocimiento de UIDPs compuestos en imágenes generadas a mano alzada.
4. Diseñar el conjunto de prestaciones o servicios para la implementación de los algoritmos basados en técnicas de *Deep Learnin*g aplicados para el reconocimiento de UIDPs compuestos.
5. Desarrollar un módulo de generación de interfaces de usuario para los dominios *e-Commerce* y *Social media* basada en técnicas *Deep Learnin*g.
6. Evaluar el módulo desarrollado a través de diferentes casos de estudio como prueba de concepto.

1.4. Justificación

Debido al impacto del crecimiento de las redes sociales y las tiendas en línea a nivel mundial, en especial las *e-Commerce y Social Media*, ha provocado que en la ingeniería de software y de manera específica en el desarrollo de software se busquen nuevas formas de disminuir

costos y mejorar los tiempos de desarrollo, tanto para el usuario final como para los desarrolladores. Partiendo de lo anterior, en esta investigación se plantea usar técnicas de *Deep Learning* para identificar UIDPs en una imagen que represente una interfaz de usuario y generar código fuente de aplicaciones *e-Commerce* y *Social Media*. El *Deep Learning* representa una solución para realizar una tarea específica y detectar diferentes tipos de la visión para un campo especial para detección de imágenes en mejorar considerablemente su calidad y desempeño.

Dicho lo anterior, los principales pilares en esta investigación se resumen en tres: el primer pilar de la investigación, es el uso de los generadores para desarrollar software para usar una imagen o un boceto generado a mano alzada, el segundo pilar es el desarrollo de software con las técnicas *Deep Learning*, que mejora considerablemente el desarrollo de software por el aprendizaje que conlleva, finalmente, el tercer pilar donde los UIDPs son parte fundamental ya que permiten solventar ciertos errores de diseño en una aplicación.

La investigación plantea una novedosa alternativa para la generación automática de código a través de imágenes o bocetos que permitan seleccionar el mejor algoritmo de *Deep Learning* para la identificación de elementos de una interfaz de usuario para aplicaciones de los dominios *e-Commerce* y *Social Media*. Las nuevas interfaces generadas se adaptan a las necesidades de más de un dispositivo sin perder calidad y por ende los usuarios mantengan una adecuada interacción. El generar código fuente por medio de las técnicas de *Deep Learning*, mediante interfaces de usuario a partir de una imagen digital proveniente de una cámara o escáner, proporciona a la ingeniería de software una alternativa más para automatizar el desarrollo de software.

Capítulo 2 Estado de la Práctica

A continuación, para este capítulo se aborda a detalle los temas de generación de aplicaciones, la interfaz de usuario y el aprendizaje profundo. En la siguiente revisión se detalla una breve reseña de cada uno de los trabajos revisados, se presenta una tabla comparativa que permite identificar las principales características que son objeto de estudio en esta investigación. Finalmente, se presenta la conclusión resultado del análisis comparativo de los trabajos estudiados.

2.1. Trabajos relacionados

En esta sección se clasificó en dos partes los trabajos relacionados, por el trabajo previo para la generación de aplicaciones o desarrollo de software y técnicas de *Deep Learning* en el desarrollo de software.

2.1.1. Herramientas para la generación de aplicaciones o desarrollo de software

Cabe mencionar que las herramientas para la generación de aplicaciones son indispensables por lo que en la literatura se reportan diferentes estudios como en Rosales-Morales et al. [24], donde analizó las herramientas para el desarrollo automático de software y la generación automática de código fuente, con el fin de evaluar y determinar si cumple o no con un conjunto de características y funcionalidades en términos de calidad. Dichas características incluyen eficacia, productividad, seguridad y satisfacción, todo a través de una evaluación cualitativa y cuantitativa. Estas herramientas son 1) herramientas CASE, 2) *frameworks* y 3) los Ambientes de Desarrollo Integrado. La evaluación se llevó acabo con el fin de medir no sólo la capacidad de uso, sino también el apoyo que brindan para el desarrollo de software automático y la generación automática de código fuente. Por último, los resultados concluyeron que Visual Paradigm es una de las mejores herramientas que contiene un buen diseño GUI (*Graphical User Interface,* Interfaz Gráfica de Usuario), una herramienta de usabilidad, que promueve el uso de características, cuenta con soporte para integración heterogénea de fuente de datos, y además tiene soporte y documentación.

Cortes-Camarillo et al. [25], realizó un análisis comparativo de patrones de diseño de interfaz de usuario (ya que facilitan el desarrollo de aplicaciones, ya que cada plataforma tiene su propia interacción entre el usuario y el dispositivo) basado en contextos de uso orientado al ámbito educativo. Se diseñó un conjunto de contextos de uso para establecer que patrones de diseño de interfaz de usuario son recomendables para un dispositivo en específico, facilitar el desarrollo de aplicaciones educativas multi-dispositivo. Por lo tanto, se propuso un diseño de interfaz de usuario de una aplicación educativa para el aprendizaje electrónico para utilizar un MOOC (*Massive Online Open Courses,* Curso en Línea Masivo y Abierto). Se aplicó el contexto de uso que se estableció y se seleccionó en los patrones de interfaz de usuario para la plataforma de dispositivo móvil. Posteriormente a partir de un bosquejo se procedió a construir la interfaz gráfica de usuario. Por lo tanto, el bosquejo facilitó el desarrollo de una aplicación educativa gracias a los beneficios que ofrecen los patrones de diseño de interfaz de usuario por los contextos de uso.

En otra investigación, Cortes-Camarillo et al. [26] planteó el diseño de una arquitectura para el desarrollo de aplicaciones educativas que se basa en un conjunto de UIDPs (*User Interface Design Pattern*, Patrones de Diseño de Interfaz de Usuario) para facilitar el desarrollo de aplicaciones laboriosos y tareas de consumo-tiempo. Como prueba de concepto se propusó EduGene, un generador de aplicaciones compatible con cuatro sistemas operativos, Android™, Firefox™ OS, macOS y Windows™ Phone, incluyendo Web. EduGene además es compatible con tres tipos de dispositivos: móviles (teléfonos inteligentes y tabletas), escritorio y televisión. Los resultados de evaluación demuestran que EduGene es un generador de aplicaciones educativo amigable, ya que provee una interfaz intuitiva que facilita la interacción del usuario, en lograr un alto puntaje en el diseño, con un contenido de la presentación y es fácil de usar. Además, logra metas educativas tales como el incremento del acceso de la información y motivar al estudiante y profesor. Las Herramientas e-learning seleccionadas para comparar el desempeño de Edugene fueron Moodel, Sakai y ATutor. Las contribuciones son: 1) el diseño en capa para los generadores de aplicaciones educativas basados en los UIDPs, 2) una representación lógica y estructurada de una aplicación

educativa a través de un documento XML y 3) el uso de los UIDPs para generar aplicaciones educativas.

Adicionalmente, Cortes-Camarillo et al. [27] presentó Atila, un generador de aplicación educativo basado en UIDPs (*User Interface Design Pattern*, Patrones de Diseño de Interfaz de Usuario). La aplicación desarrollada soporta cuatro sistemas operativos: Android™, FireFox™ OS, macOS™, Windows™ Phone y Web. Ya que los UIDPs facilitan la interacción entre la aplicación y el usuario, además provee la oportunidad de generar software de alta calidad. Como resultados se propuso generar una aplicación para las plataformas macOS™ y Windows™ Phone, primero se necesita activar y desactivar las características requeridas para la aplicación a desarrollar. Segundo, se hace una selección de las opciones en la herramienta y se genera un archivo XML. Tercero se descarga el proyecto generado. Para generar la aplicación, solamente se necesita indicar el nombre de la UIDP y su posición en la aplicación. Atila permite al usuario modificar libremente la aplicación del diseño de interfaz de usuario. Otro de los beneficios es la velocidad y la facilidad para desarrollar aplicaciones educativas para dispositivos móviles a través del uso de UIDPs.

Sánchez-Morales et al. [28] presentaron un componente de software para generar interfaces de usuario para aplicaciones móviles en usar el reconocimiento de patrones, procesamiento de imagen, y técnicas de redes neuronales. El proceso de generación de interfaces de usuario consiste en tres fases: 1) análisis de la imagen, 2) configuración y 3) generación de código fuente. Partiendo de esto, novedosos y nuevos enfoques para procesos de desarrollo para aplicaciones móviles requeridos de modo que las técnicas de inteligencia artificiales aplican el propósito de agilizar, facilitar e interactuar desarrollos a través de métodos computacionales que involucren redes neuronales, procesamiento de imagen digital, y técnicas de reconocimiento de patrones. Como resultado se presentó la prueba del módulo mediante una imagen a mano alzada que representa una interfaz de usuario para transformar al código fuente de una aplicación móvil. El porcentaje de eficiencia en reconocimiento de elementos de la interfaz usada para el caso de estudio fue de 94.3% Por lo tanto, la

identificación elementos de interfaz de usuario en una imagen generada a mano alzada a través de redes neuronales es totalmente posible.

Da-Costa et al. [29] aplicó un reciente MDD (*Model-Driven Development*, el Desarrollo Dirigida por Modelos), que es un paradigma para brindar el reúso y ocupar las brechas de productividad, por medio de modelos abstractos y la generación automática de software a través de transformaciones de modelos para avanzar en la investigación de la UI (*User Interface,* Interfaz de usuario), dicho lo anterior, se propuso el estereotipo UI para la ingeniería UI en el dominio de sistema Web. El estereotipo de la UI captura los siguientes puntos: 1) las especificaciones de la UI, 2) el modelado recurrente de la presentación de la UI y 3) el comportamiento de las interacciones y tareas del usuario. Se aplica este concepto para describir un estereotipo de un portal web de la UI como un patrón de interacción recurrente que permite la generación automática de muchos componentes basados en prácticas impulsadas por el modelo para portales Web de la UI. Este enfoque de generar la UI es compatible con recientes avances en la construcción de la UI, tales como la interacción IFML (*Interaction Flow Modeling Language*, Lenguaje de Modelado de Flujo de Interacción), es un reciente modelo de la UI estandarizado por la OMG. Los resultados obtenidos de los portales web de la UI reducen el desarrollo de software time-to-market, esfuerzo y costo, contribuyendo ambos en la calidad y en el mejorar la mantenibilidad de las aplicaciones Web.

Sánchez-Morales et al. [30], propuso el desarrollo de un componente de software que utiliza técnicas de procesamiento de imágenes y reconocimiento de patrones para la generación automática de aplicaciones multi-dispositivo. Ya que existen diferentes esquemas de generación automática de software como son MDA (*Model Driven Architecture*, Arquitectura Dirigida por Modelos), FDD (*Feature-driven Development*, Desarrollo basado en funcionalidades) y RAD (*Rapid Application Development*, Desarrollo rápido de aplicaciones). Sin embargo, hasta el momento no se aborda la generación de software mediante técnicas de inteligencia artificial y reconocimiento de patrones. Por lo tanto, para lograr ese objetivo se propuso identificar elementos de una imagen y generar el código de

una aplicación multi-dispositivo con los elementos detectados previamente seleccionados en dicha imagen. Como resultado se planteó un caso de estudio como prueba de concepto del módulo, se representó la generación de código de la portada principal de una página Web a partir de la presentación de la interfaz en una imagen. Los autores concluyeron que la aplicación de un componente de software facilita la tarea de crear prototipos de aplicaciones para presentarlas al usuario antes de iniciar un desarrollo, permitiendo ahorrar tiempo y dinero.

2.1.2. Técnicas *Deep Learning* en el desarrollo de software

Dentro del desarrollo del software el uso de técnicas de *Deep Learning* son importantes y tienen diversas aplicaciones. A. Halbe y Dr. A. R. Joshi [31] sugirieron un enfoque novedoso para crear automáticamente páginas HTML (*HyperText Markup Language*, Lenguaje de Marcas de Hipertexto) por medio de un diseño GUI (*Graphical User Interface*, Interfaz Gráfica de Usuario) dibujado sobre el papel considerado como un diseño GUI, ya que contiene varios controles HTML tales como radio button, checkbox, textbox, command button y label. Al escanear una imagen de un diseño GUI esta se ingresa al sistema, el sistema segmenta la imagen en varios controles y el segmento de control HTML es identificado. Una vez identificado los controles estos se almacenan en una base datos en XML. El XML contiene el nombre y la posición del componente sobre el diseño de la GUI, y este archivo XML es analizado para generar una página HTML. En conclusión, los resultados obtenidos de un caso de estudio recopilan datos empíricos, para ello se procesan y escanean 30 diseños GUI dibujados a mano en diferentes técnicas propuestas. Por último, se concluye que los diseños GUI permite ser convertidos a páginas funcionales HTML con bastante exactitud.

En Baveye et al. [32] se presentó un modelo refinado computacional de aprendizaje profundo para la predicción de la memorabilidad de la imagen, el desempeño de este modelo es superior significativamente y obtiene relativamente un 32.78% de incremento en comparación a los modelos mencionados en el estado del arte sobre el mismo conjunto de datos. Además, se investigó el modelo que generalizó un nuevo conjunto de datos de 150 imágenes, para cada memorabilidad y la investigación del puntaje efectivo que recolectaron

más de 50 participantes. Dicho lo anterior el rendimiento de la predicción fue más débil sobre este nuevo conjunto, destaca el problema de la representatividad en los conjuntos de datos. En particular, el modelo obtuvo un gran desempeño de predicción para imágenes negativas activas que las imágenes neutras o imágenes positivas activas, el recodar qué tan importante es el conjunto de memorabilidad en la consistencia de imágenes que estén adecuadamente distribuidas dentro del espacio emocional. Los resultados sugirieron que al tomar en cuenta el efecto en el conjunto de datos de imágenes etiquetadas como las puntuaciones de memorabilidad para asegurarse que se induce una gran variedad de emociones, de hecho, la información emocional y la memorabilidad se encuentran relacionadas, el desempeño de un modelo de memorabilidad depende de la información emocional inducida por las imágenes utilizadas para entrenar el modelo. También se sugirió que la información emocional, es una valiosa cualidad para aumentar el rendimiento del modelo para imágenes neutrales y positivas, ya que infiere computacionalmente la información emocional.

Por otra parte, en Bianco et al. [33] se planteó un método para el reconocimiento de logos en imágenes o videos para aplicar el aprendizaje profundo. El reconocimiento está compuesto por una región localizada del logo seguido por una CNN (*Convolutional Neural Network*, Red Neuronal Convolucional) específicamente entrenada para la clasificación del logo, incluso si no están localizados con precisión. Los experimentos del método ocuparon la base de datos FlickrLogos-32, y se evaluaron los efectos del desempeño del reconocimiento del aumento de datos sintéticos contra los datos reales y el pre-procesamiento de imagen. Además, se investigaron los beneficios y las diferentes elecciones de los entrenamientos, como son: las clases balanceadas, ponderación de la muestra y modelado explícito de la clase de fondo (la región que no contiene logo). En conclusión, los resultados del experimento confirmaron la factibilidad del método propuesto que mejora los métodos presentados en el estado del arte. Otro punto a comentar, es que el reconocimiento del logo es fundamental en muchos dominios de aplicación, el problema recurrente es que el logo sobre la imagen tiende a aparecer en diferentes posiciones y escalas o sobre cualquier otro lugar de la vista de la imagen. Incluso al reconocer el logo sobre las imágenes susceptibles a ser corrompidas por diferentes mecanismos de la imagen y distorsiones.

Por su parte, Liu et al. [34] presentó una revisión sistemática de la literatura de fusión de imágenes a nivel de pixel basada en el aprendizaje profundo. Específicamente, se detallaron las principales dificultades que existen en la investigación de fusión de imágenes convencionales y se discutieron las ventajas que el aprendizaje profundo ofrece para direccionar cada uno de estos problemas. De tal modo que, los recientes logros de la fusión de imágenes basado en el aprendizaje profundo es revisado a detalle. Actualmente, más de una docena de métodos propuestos de fusión de imágenes basadas y analizadas en técnicas de aprendizaje profundo incluyendo las CNN (*Convolutional Neural Networks*, Red Neuronal Convolucional), CSR (*Convolutional Sparse Representation*, Representación dispersa Convolucional) y SAE (*Stacked Autoencoders*, Codificadores Automáticos Apilados). Por último, los existentes métodos de fusión de imágenes basados en aprendizaje profundo en varios *framewoks* genéricos y en presentación de un potencial *framework* basado en aprendizaje profundo para desarrollar métricas de evaluación objetivas. Las contribuciones desarrolladas son las siguientes: 1) las pertinentes dificultades se resumen dentro de cuatro puntos específicos en términos de métodos de fusión y métricas de evaluación objetiva; 2) se presentan las ideas básicas, los pasos principales, las aplicaciones específicas y las características principales de los métodos; 3) los principales problemas y retos que existen en cada *framework* se discuten.

En Pang et al. [35] se expuso un nuevo enfoque para hacer frente a tareas de seguimiento de objetivos visuales. El método usa una red neuronal convolucional que permite clasificar una serie de parches dependiendo de qué tan bien se encuentra centrado. Para cubrir las posibles interferencias se pretende entrenar la red con parches ubicados sobre el entorno detectado del objeto centrado, y con diferentes tamaños, así toma en cuenta cambios eventuales de escala. Por otro lado, al entrenar la red, se crean un conjunto de datos ad-hoc con ejemplos positivos y negativos para la extracción del objeto centrado desde la base de datos de detección de Imagenet. Los ejemplos positivos son aquellos que tienen el objeto centrado correctamente, mientras que los negativos son aquellos que están centrados incorrectamente. Finalmente, se seleccionan los parches más prometedores, al usar la función de emparejamiento basado en

las características de profundidad que es proveída por la red AlexNet. Por lo tanto, es rápido y útil para el seguimiento visual en tiempo real. Los resultados muestran que los métodos son muy competitivos con respectos a los algoritmos descritos en el estado del arte, siendo muy robustos contra interferencias típicas durante el proceso de seguimiento de objetivos visuales.

Krishevsky et al. [36] entrenaron una gran red neuronal convolucional para clasificar 1.2 millones de imágenes de alta resolución en ImageNet LSVRC-2010 en contienda de 1000 diferentes clases. Sobre los datos de prueba, se archivaron las tazas de error de top-1 y top-5 de 37.5% y 17.0% que es considerablemente muy bueno en lo mencionado en el estado del arte. La red neuronal, cuyos parámetros son de 60 millones y 650,000 neuronas, consiste de 5 capas convolucionales, algunos de los cuales están compuestos por capas max-pooling y tres capas totalmente conectadas con una 1000-way softmax. Además, para entrenar con rapidez, se usaron neuronas no saturadas y un eficiente GPU para la operación de convolución. Para reducir el sobreajuste en las capas totalmente conectadas se empleó un método llamado "dropout" que es muy efectivo. Asimismo, se ingresó una variante de la competencia ILSVRC-2012 obteniendo una tasa de error de prueba entre los top-5 con 15.3% en comparación con los 26.2% de resultado dado por la segunda revisión del entrenamiento. Como conclusión se demuestra que una gran profunda red neuronal convolucional es capaz de lograr resultados récord para un conjunto de datos altamente desafiantes en utilizar puramente el aprendizaje supervisado.

Huang et al. [37] desarrollaron técnicas de recuperación de imagen para imágenes de base de datos Web de gran escala. Dicho lo anterior, es un avanzado sistema de recuperación, denominado MRBDL (Multi-concept Retrieval using Bimodal Deep Learning, Multi-concepto de Recuperación al usar Aprendizaje Profundo Bimodal) es propuesto e implementado el uso de las redes neuronales convolucionales, que efectivamente guardan correlaciones semánticas entre imágenes visuales y etiquetas de contextos libres. La cual difiere de los enfoques existentes que utilizan conceptos múltiples e independientes en una consulta. MRBDL considera múltiples conceptos como una escena holística para el aprendizaje modelo de recuperación. Particularmente, se usa en una red neuronal

convolucional bimodal para entrenar una escena de clasificación holística en dos modalidades, y entonces una correlación semántica de inclusión de sub-conceptos en las imágenes que fuerzan el impulso del reconocimiento de la escena holística. El récord de predicción semántica obtenido desde la clasificación de la escena holística combinada con información complementaria de imágenes Web para mejorar el desempeño de recuperación. Los resultados detallaron que el enfoque propuesto mejora favorablemente a comparación con diferentes métodos del estado del arte.

Nedzved et al. [38] proponen un esquema para desarrollar procesamiento de imágenes y el análisis de software basado en la generación de tablas de tesauros en la interpretación del núcleo. El esquema está basado en la combinación de características de bibliotecas dinámicas y de un intérprete con un conjunto de funciones para el procesamiento de imagen. Como resultado, el programa se divide en dos partes: el primero apunta a un desarrollo de software profesional y el segundo a usuarios. Por último, se propone un procesamiento histológico de imágenes por especie donde la tecnología de desarrollo de software basado en generación automática de letras, incluye un conjunto de funciones de procesamiento de imágenes simples para diferentes problemas de análisis de tejido histológico. El propósito de la arquitectura del software facilita el desarrollo de programas para análisis de imagen medico en particular, análisis histológicos.

2.2.Análisis comparativo

En la siguiente tabla, el análisis se realizó al considerar los siguientes aspectos: dominio, generación de código, el soporte de UIDPs, multidispositivo y el uso del *Deep Learning.* El aspecto de dominio se encuentra enfocada al ámbito de aplicación del artículo, la generación de código fuente se refiere a si dentro del trabajo se soporta la generación de código fuente, la característica de soporte de UIDPs hace referencia al uso de los patrones de diseño, respecto a la característica multidispositivo, se refiere a si en el trabajo se toma en cuenta la compatibilidad de diferentes dispositivos. Finalmente, el uso del *Deep Learning* se refiere a la aplicación de alguna de estas técnicas conocidas.

Tabla 2.1 Análisis comparativo del estado del arte

Autor	Dominio	Generación de Código Fuente	Soporta UIDPs	Multidispositivo	Uso del Deep Learning
Rosales-Morales et al., 2015 [24]	Ingeniería de Software	Sí	No	Sí	No
Cortes-Camarillo et al., 2016 [25]	UIDPs	No	Sí	Sí	No
Cortes-Camarillo et al., 2018 [26]	UIDPs	Sí	Sí	Sí	No
Cortes-Camarillo et al., 2017 [27]	Ingeniería de Software	Sí	Sí	Sí	No
Sánchez-Morales et al., 2017 [28]	UIDPs	Sí	Sí	Sí	No
Larissa-da-Costa et al., 2014 [29]	Ingeniería de Software	Sí	No	Sí	No
Sánchez-Morales et al., 2015 [30]	Multidispositivo	Si	No	Sí	No
Halbe& Joshi, 2015 [31]	Procesamiento de imágenes para GUI, IA	Sí	No	Sí	No
Baveye et al., 2015 [32]	I.A.	No	No	Sí	Sí
Bianco et al. 2017 [33]	I.A.	No	No	Sí	Sí
Liu et al., 2018 [34]	I.A.	No	No	Sí	Sí
Pang et al., 2017 [35]	I.A.	No	No	Sí	Sí
Krizhevsky et al., 2012 [36]	I.A.	No	No	Sí	Sí
H2uang et al., 2017 [37]	I.A.	No	No	Sí	Sí
Nedzved et al., 2013 [38]	I.A.	No	No	Sí	No

Como conclusión se observa que los artículos revisados no consideran todas las características analizadas al mismo tiempo. Todos los trabajos soportan el desarrollo multidispositivo, pero no necesariamente permiten la generación de código fuente, o la implementación de UIDPs en el desarrollo de sus interfaces, y además el uso de técnicas de *Deep Learning* como parte de las técnicas en el proceso de desarrollo. Por tal motivo se observa la oportunidad de abordar estas técnicas para plantear un proceso de desarrollo de software basado en técnicas de *Deep Learning*, que parta del uso de imágenes generadas a mano alzada que representen interfaces de usuario.

2.3. Solución propuesta

En este punto se presenta la propuesta de solución para el desarrollo de la investigación, el valor que representa la solución está sustentado en desarrollo confiable, gratuito y mantiene estándares de desarrollo óptimos. En la tabla 2.2 se muestra la propuesta de la solución que se seleccionó.

Tabla 2.2 Análisis de la propuesta de solución

Propuesta	
	Solución
Lenguaje de Programación	Python
Servidor de Aplicaciones	Apache HTTP Server
Metodología para el desarrollo de software	Prototipado
Entorno de desarrollo	Anaconda, Spyder

De acuerdo a la solución propuesta de la tabla 2.2 y partiendo directamente del uso de técnicas de *Deep Learning* hay información favorable y concentrada para el lenguaje de programación en Python, por lo que se toma como primera opción Python como lenguaje de programación. Además, se propone un servidor Apache HTTP Server en caso de requerir ajustar detalles del servidor. Se integra también la metodología Prototipado cuyas cualidades

son principalmente, mantener calidad, disciplina y retroalimentación continúa durante el proyecto. Y por último se encuentra el entorno de desarrollo compuesto por la plataforma de Anaconda, que dispone del IDE Spyder, una herramienta muy versátil con características especiales a favor de las técnicas y modelos de datos científicos.

2.4. Justificación de la solución propuesta

La solución propuesta de la tesis considera tecnologías habituales como también las más factibles, es necesario comprender que el lenguaje de programación tiene que ser una opción que más se adapte a las necesidades de las técnicas y sustente de información a las API de *Deep Learning*. De acuerdo a la propuesta elegida, se justifica la elección de Python, que es un lenguaje base para el desarrollo de técnicas y algoritmos de *Deep Learning*, provee una documentación más específica a la hora de programar dicha técnica, su comunidad y la facilidad del lenguaje de programación provee un sin fin de oportunidades a desenvolver en las técnicas de *Deep Learning*. Por otro lado, como servidor de aplicaciones se elige Apache, porque permite la edición de código fuente, cuenta con la ventaja de permitir el desarrollo de su propio núcleo o la ampliación de otras funciones. Apache es portable lo que permite usarlo en diferentes sistemas operativos, además el costo que se maneja resulta económico o maneja la licencia de software libre, mantiene una amplia contribución y colaboración de usuarios con un gran soporte técnico, suma una gran variedad extensiones y con soporte para el lenguaje Python. La elección de la metodología prototipado es un punto importante ya que como es un proceso iterativo, prototipado ya que no se conocen los objetivos generales y en caso no se conoce la eficacia de un algoritmo esto ayudaría en gran parte, ya que una red neuronal puede variar por las características de entrenamiento, de cantidad de instancias y configuración Como punto final, se aplicó en la plataforma Anaconda el entorno de desarrollo de Spyder por dos motivos, tiene una gran variedad de extensiones que proveen una compatibilidad con el lenguaje en curso y además se obtiene un repositorio de bibliotecas y herramientas para un desarrollo eficiente, tales como: la cobertura de código fuente, análisis de código, entre otras.

Capítulo 3 Aplicación de la metodología

A continuación, se presenta una descripción de cómo está conformado el proceso generación, revisión de los UIDPs para los dominios *e-Commerce y Social Media*, al final se presentan algunos casos de uso que se implementaron como prueba de concepto del proceso y técnicas implementadas.

3.1.Proceso de generación de software

En esta sección se aborda claramente el proceso de generación automática de aplicaciones basado en técnicas de *Deep Learning* para la identificación de UIDPs que determinan el desarrollo de una aplicación para *e-Commerce* y *Social Media*. En la figura 1 el proceso consta de tres etapas principales: análisis de la imagen, la configuración y posteriormente la generación de código. A continuación, se describen cada una de las etapas.

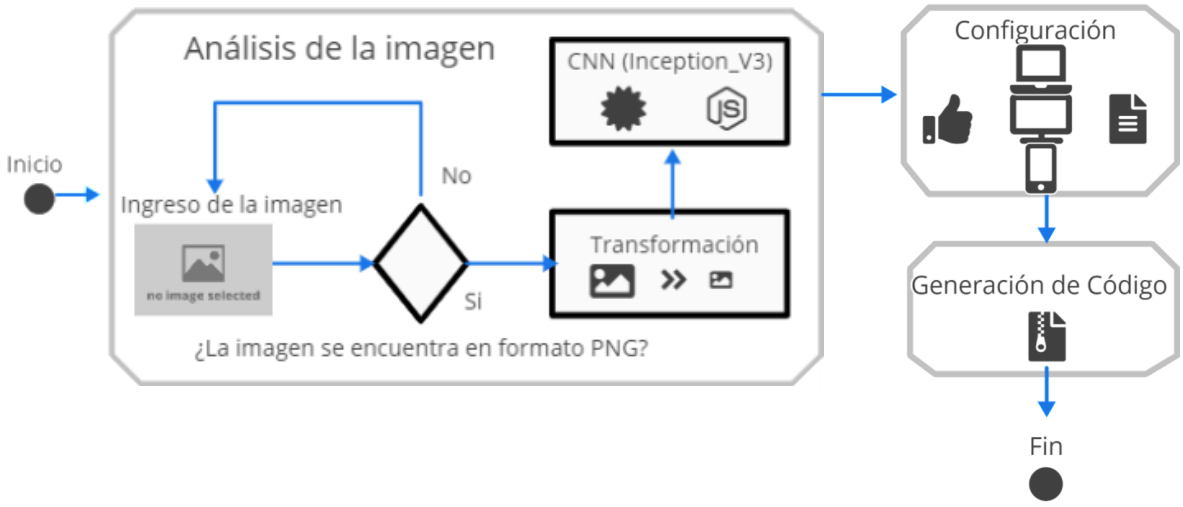


Fig. 3.1 Proceso de generación de software a partir de técnicas Deep Learning.

Análisis de la imagen: esta etapa considera la realización de un análisis de las imágenes que soportan los métodos para posteriormente generar el código fuente de aplicaciones de *e-Commerce* y *Social Media*.

1. Entrada de la imagen a procesar: las imágenes soportadas son interfaces de usuario dibujadas a mano alzada y digitalizadas a través de una cámara fotográfica o escáner y

cargadas en memoria. Hasta el momento se soporta solo el formato PNG dado que permite una mayor calidad de los datos. Las imágenes tienen dimensiones de 1050x650 pixeles. Es importante mencionar que el algoritmo no está limitado a esta única resolución de imágenes, es posible adaptar el algoritmo para diferentes resoluciones de forma automática. Sin embargo, para fines de pruebas se aplicó una única resolución.

2. Validación de la imagen de entrada: la validación de la imagen consiste verificar que el formato de la imagen sea PNG. Si el formato es correcto, se procede con el paso de transformación.

3. Transformación: Este paso consiste en aplicar un conjunto de algoritmos de procesamiento de imágenes con el objetivo de extraer y verificar que al menos un elemento en la imagen de entrada representa a un Patrón de Diseño de Interfaz de Usuario. Es importante mencionar que la transformación permite mejorar la identificación de elementos en la imagen. Para lograrlo, el modelo de la red neuronal convolucional aplica una transformación a la imagen que permite eliminar sombras y ruidos de la imagen para que después el modelo clasifique los elementos encontrados en la imagen.

4. Identificación de elementos: a través de los algoritmos de Deep Learning como Inception_V3 se obtienen distintos componentes de la imagen. Los componentes son representaciones de patrones de diseño de interfaz de usuarios, y además válidos para el desarrollo de aplicaciones e-Commerce y Social Media. Los algoritmos de Deep Learning se implementan a través de una red neuronal convolucional. El proceso de la red neuronal convolucional consiste en tomar un núcleo de la imagen para la siguiente capa y clasificar, así consecutivamente hasta la clasificación completa del patrón en la capa de salida. El modelo de Inception_v3 es una red neuronal que satisface la clasificación de las imágenes y muestra en formato JSON los elementos identificados ordenados por porcentajes desde el más alto hasta el más bajo, para este punto se toman los cuatro primeros porcentajes más cercanos a cada UIDP.

Configuración: el objetivo principal de esta etapa es, permitir la configuración del tipo de aplicación a generar en este caso, e-Commerce y/o Social Media; después de la identificación de los UIDPs en la imagen durante el proceso anterior. Los pasos de este proceso son los siguientes.

1. Selección del dominio: la selección del dominio de la aplicación a generar está basado en los UIDPs identificados en la etapa anterior, ya que no todos los UIDPs serán válidos para los dominios e-Commerce y Social Media. El tipo de aplicaciones e-Commerce soportadas son B2C Y C2C mientras que para Social Media se consideran aplicaciones de Microbloggin, Streaming, y Redes sociales.

2. Selección del tipo de aplicación a generar: En este caso, se permite generar aplicaciones Web, aplicaciones para dispositivos móviles e inclusive de TV. Esta decisión se toma con la finalidad de que la generación de código se realice según las características de cada una de estas aplicaciones. Para la selección del tipo de aplicación se toma en cuenta los UIDPs identificados, ya que no todos los UIDPs son válidos para cualquier dispositivo y/o plataforma.

3. Configuración de la aplicación: de acuerdo con el dominio y tipo de aplicación seleccionada en los pasos previos, se selecciona la configuración general de la aplicación. Algunos de los datos de configuración incluyen título principal, una cadena compuesta para 50 caracteres; lenguaje de desarrollo una cadena de 20 caracteres, los lenguajes soportados son para Web con HTML5, Android® con Java y iOS® con Objetive –C. Otro dato de configuración es la versión de la aplicación, donde se aceptan solo números, puntos y guiones. Finalmente, datos del autor de la aplicación como nombre, apellidos, *e-mail*, empresa y *e-mail* de la empresa.

Generación de código: durante esta etapa se realiza la transformación de la información recolectada a lo largo del proceso de identificación y configuración necesarios para generar el código fuente de las aplicaciones. Este proceso consta de un conjunto de pasos necesarios para lograr su objetivo, los pasos se describen a continuación.

1. Generación de un documento XML: se genera un documento XML que contenga etiquetas para almacenar el conjunto de datos obtenidos durante la configuración, dichos datos serán: nombre de la aplicación, autor (s), etiquetas con la configuración de acuerdo a cada tipo de aplicación y lenguaje seleccionado, resolución, orientación y elementos de plantilla (*header, body, footer, left y right*). Finalmente, un conjunto de etiquetas con la representación de cada uno de los elementos identificados en la imagen.

2. Transformación del documento XML a código fuente: la transformación consiste primero en procesar el documento XML por medio de un documento XML Schema con el objetivo de validar la estructura y garantizar la correcta generación del código fuente de la aplicación. Si la estructura es correcta se genera el equivalente en código por plataforma de los UIDPs descritos en el documento XML. La estructura generada se agrupa en clases y se almacenan en una carpeta por plataforma de acuerdo con el lenguaje de programación seleccionado por el usuario.

3. Archivo ZIP con el código fuente generada: con el código fuente generado se empaquetan las carpetas con los proyectos generados por plataforma y se crea un único archivo ZIP, el usuario descarga el código fuente que se modifique o adecue conforme a sus requerimientos y como mejor le convenga. En este sentido, los proyectos generados tienen la ventaja de exportarse a entornos de desarrollo de acuerdo a la plataforma de desarrollo. Los entornos de desarrollo válidos para exportar los proyectos son Dreamweaver u otro entorno de desarrollo similar en HTML5; para iOS® el entorno de Xcode™ y por último Android StudioTM para el proyecto Android.

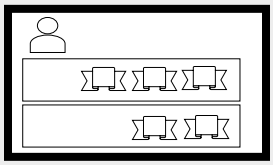
3.2. Elementos identificados de UIDPs

En esta sección se identifican y analizan los UIDPs adecuados y válidos para el desarrollo de aplicaciones *e-Commerce* y *Social Media*. El análisis consistió en seleccionar un conjunto de aplicaciones de *e-Commerce y Social Media* estas aplicaciones elegidas por la mayor calificación dada por los usuarios, además de ser las más usadas, con el objetivo de identificar los UIDPs más cualificados en cada aplicación.

3.2.1. Análisis de UIDPs en *Social Media*

En este punto se presentan los resultados del análisis de los UIDPs del dominio *Social Media*. En este sentido, se identificaron dos tipos de UIDPs y se dividen en UIDPs por reputación e interacción. En la Tabla 3.1 se describen los UIDPs por reputación, estos permiten construir prestigio a los usuarios mediante el contenido, información compartida, la relación con más personas y seguir sus actividades.

Tabla 3.1 UIDPs Social Media por reputación

Reputación	Descripción	Imagen
Leaderboards	Se usan para mostrar usuarios altamente competitivos de las actividades que desarrollan.	
Achievements	Se aplica para desplegar las metas o el logro alcanzado en alguna actividad y compartirlo con tu círculo de amigos.	

En la siguiente figura se muestra la representación real de los UIDPs de tipo Social Media: *leaderboards* y *achievements* (ver Fig. 3.2).

Leaderboards *Achievements*

Fig. 3.2 UIDPs Social Meda por reputación: leaderboards y Achievements

En la Tabla 3.2 se listan los UIDPs por interacción social, los cuales permiten compartir información, conectarse con más personas y seguir sus actividades.

Tabla 3.2 UIDPs Sociale Media por interacción

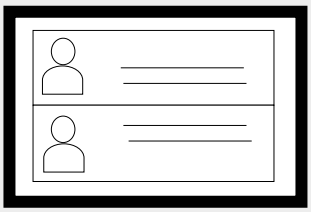
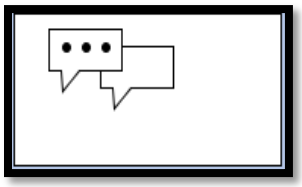
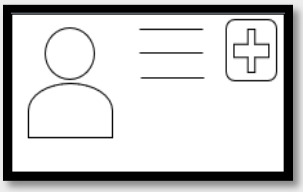
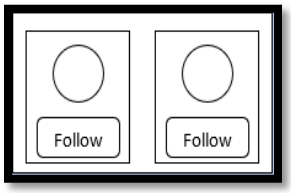
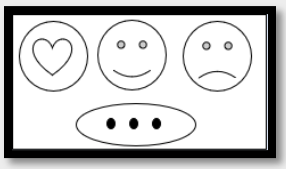
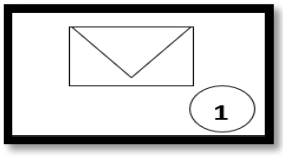
Interacciones Sociales	Descripción	Imagen
Frient list	Ofrece una manera de explorar a los usuarios de tu círculo cercano, da oportunidad de encontrar contactos frecuentes o la forma de compartir la información con los demás.	
Chat	Permite a los usuarios contactar con otros usuarios de su mismo círculo o externos y provee una forma de comunicación privada de persona a persona.	
Friend	Permite ver la información del contacto, como funciones importantes es enviar amistad y hablar información	
Follow	Permite seguir al contacto seleccionado además de ver las ultimas actualizaciones de sus actividades.	
Reaction	Permite calificar o realizar un cambio producido bajo un estímulo del usuario sobre el contenido de usuarios	
Invite friends	Permite invitar a más amigos a consultar o visitar un sitio o información específica.	

Tabla 3.3 UIDPs Sociale Media por interacción

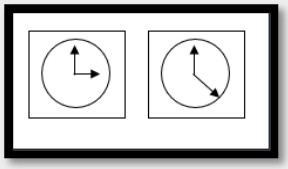
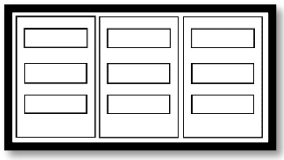
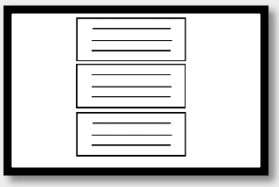
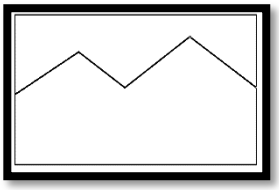
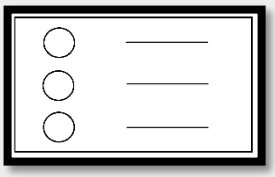
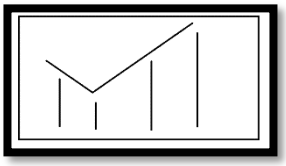
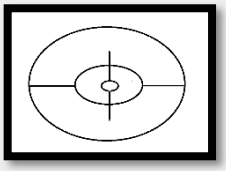
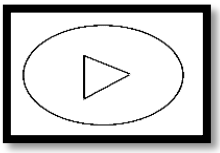
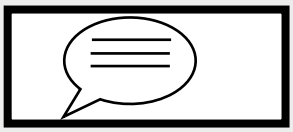
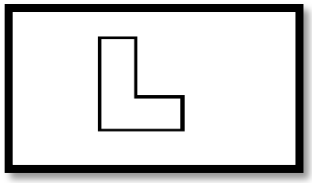
Interacciones Sociales	Descripción	Imagen
Activity Stream	Actualiza las actividades de los contactos que respondan a las acciones como reacciones y recordatorios	
Horizontal Menu	Muestra en forma de lista las opciones de la aplicación en forma horizontal.	
Vertical Menu	Muestra en forma de lista las opciones de la aplicación en forma vertical.	
Gallery	Permite ver imágenes de modo secuencial, de principio a fin o cuando se desea ver una imagen lo más posiblemente reservada de resolución.	
Panel	Permite ver las características o propiedades del aspecto, como un producto o actividad.	
Ranking	Permite valorar o calificar por parte de un usuario un aspecto que contenga la aplicación.	

Tabla 3.4 UIDPs Sociale Media por interacción

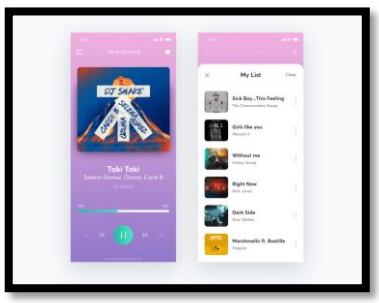
Interacciones Sociales	Descripción	Imagen
Picker	Permite seleccionar los ajustes de un calendario, paleta de color, entre otros.	
Music Player	Reproduce música y sonidos seleccionados.	
Comment	Permite escribir memorias o historias con brevedad sobre algún contenido.	
Login	Permite ingresar un nombre de usuario y una contraseña para iniciar sesión.	

A continuación, se muestra la representación real de UIDPs de *Social Media*: *Horizontal Menu*, *Ranking*, *Music Player* y *Login*, *Gallery*, *Reaction y Friend* (ver Fig. 3.3 y 3.4).

Menu-Horizontal

Ranking

Music Player



Login

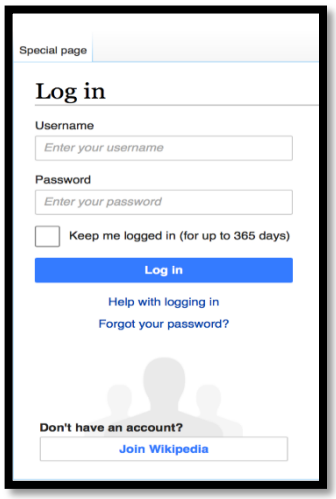


Fig. 3.3 contiene la representación de: *Menu-Horizontal, Ranking y Music Player y Login*

Gallery *Reaction* *Friend*

Fig. 3.4 UIDPs Social Meda por Interacción: Gallery, Reaction y Friend

En la Fig. 3.5 se representa ejemplos de los UIDPs: *Comments*, *Follow* y *Picker*.

Comments *Follow* *Picker*

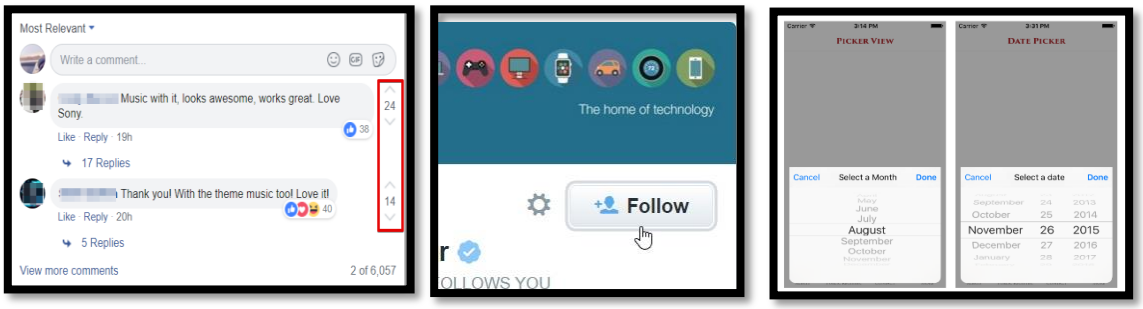


Fig. 3.5 UIDPs Social Meda por Interacción: Comments, Follow y Picker

La Fig. 3.6 se cuenta con la representación real de *Chat*, *Vertical Menu*, y *Activity Stream*.

Chat *Menu-Vertical* *Activiy Stream*

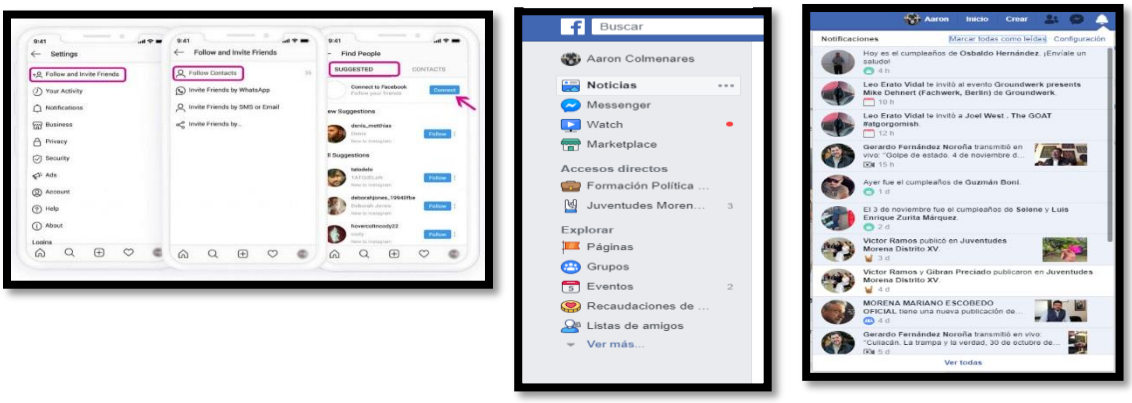


Fig. 3.6 UIDPs Social Meda por Interacción: Chat, Menu Vertical y Activity Stream

Por último, en la Fig. 3.7 se representan ejemplos de los UIDPs: *Friend List* y *Panel*.

Friend List *Panel*

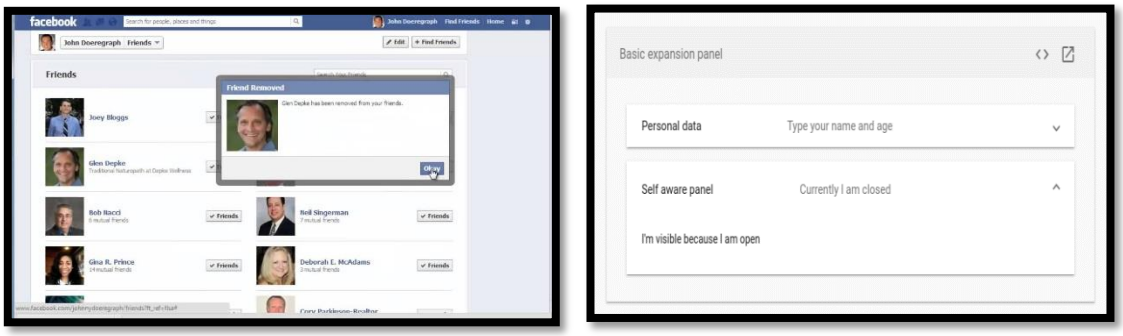


Fig. 3.7 UIDPs Social Meda por Interacción: Friend List y Panel

3.2.2. Análisis de UIDPs en *e-Commerce*

En la tabla 3.3 se describen los UIDPs para las aplicaciones de tipo e-Commerce, donde este tipo de UIDPs permiten al usuario interactuar con el producto que se selecciona o pretende comprar.

Tabla 3.5 UIDPs e-Commerce

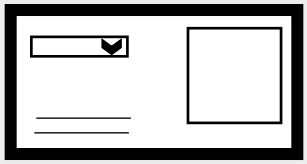
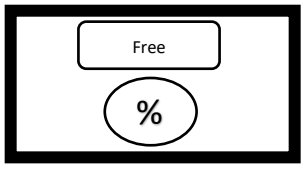
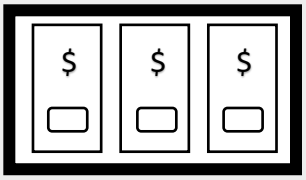
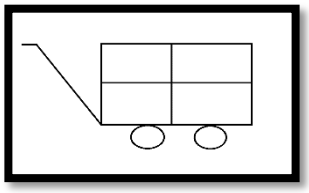
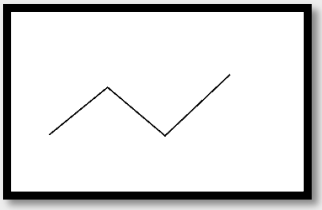
Reputación	Descripción	Imagen
Product Page	Muestra información en un sitio comercial sobre un producto y la disponibilidad de este.	
Coupon	Crea atención a un producto o servicio, incentiva a los clientes y anuncia los descuentos.	

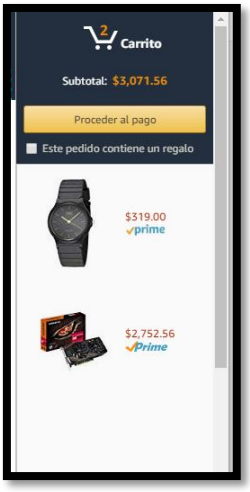
Tabla 3.6 UIDPs e-Commerce

Reputación	Descripción	Imagen
Pricing table	Permite dar información de sus características y precios de uno o más versiones de un producto.	
Shopping Cart	Permite comprar uno o más productos, además de almacenar la compra para continuar posteriormente	
Stats	Permite ver las estadísticas de valores estáticos o dinámicos	

La figura 3.8 muestran de representaciones reales de los UIDPs: *Coupon*, *Shopping Cart* y *Pricing Table*.

Coupon

Shopping Cart



Pricing Table

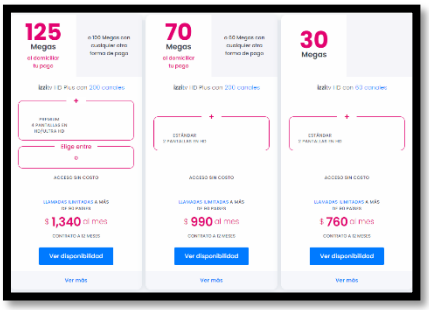


Fig. 3.8 UIDPs e-commerce: Coupon, Shopping Cart y Pricing Table

En la Fig. 3.9 se observa la representación real de los UIDPs: *Product Page* y *Stats*.

Product Page *Stats*

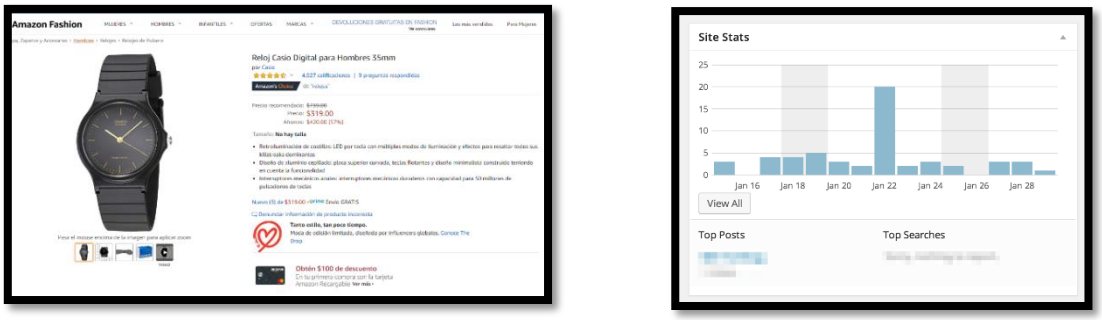


Fig. 3.9 UIDPs e-Commerce: Product Page y Stats

Los patrones revisados para las aplicaciones *e-Commerce* y *Social Media* permite conocer más afondo la estructura y funcionalidades principales que permiten interpretar de forma correcta el desarrollo de aplicaciones de tipo *e-Commerce* y *Social Media* a través del adecuado uso de UIDPs en el diseño de interfaces de usuario que garantice su buena aplicación en la vida real.

3.2.3. Análisis de Composiciones de UIDPs

En este punto se presenta el análisis de las composiciones generadas por las UIDPs simples que se presentan en la sección 3.2.1 y 3.2.2 organizadas por dominio: *e-Commerce* y *Social media*. En la Tabla 3.4 se especifican los detalles de cada composición de tipo *e-Commerce.*

Tabla 3.7 Composiciones de UIDPs para e-Commerce

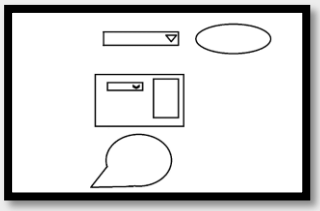
Composición	Descripción	UIDPs	Icono
Bus-men-pro-com	Aplicación que permite mostrar información de uno o más productos	*Search* *Menu* *Product page* *Comment*	

Tabla 3.8 Composiciones de UIDPs para e-Commerce

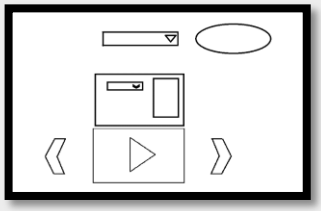
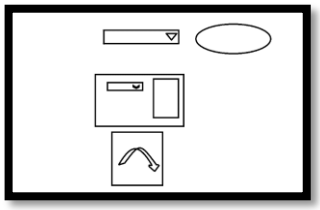
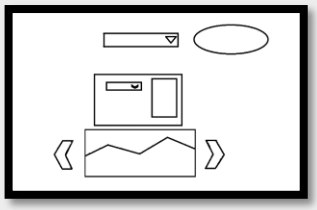
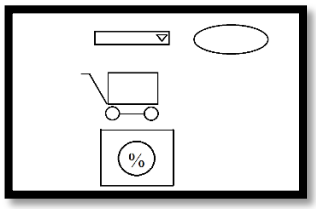
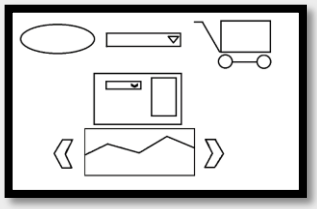
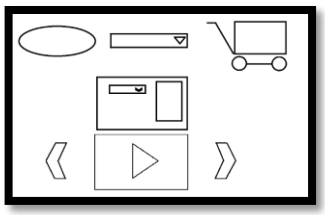
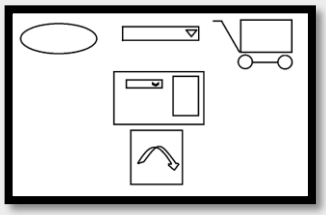
Composición	Descripción	UIDPs	Icono
Bus-men-pro-pla	Aplicación de información del producto con reproducción multimedia	*Search* *Menu* *Product page* *Player*	
Bus-men-pro-tip	Aplicación de información del producto que permite compartir la información	*Search* *Menu* *Product page* *Tip a friend*	
Bus-men-pro-gal	Aplicación de información del producto con distintas imágenes de un producto	*Search* *Menu* *Product page* *Gallery*	
Bus-men-sho-cou	Aplicación para la compra de un producto y para aplicar descuento	*Search* *Menu* *Shopping Cart* *Coupon*	
Bus-men-sho-pro-gal	Aplicación para la compra de un producto y con distintas imágenes	*Search* *Menu* *Shopping Cart* *Product page* *Gallery*	
Bus-men-sho-pro-pla	Aplicación para comprar un producto y visualizar los archivos multimedia	*Search* *Menu* *Shopping Cart* *Product page* *Player*	
Bus-men-sho-pro-tip	Aplicación para comprar un producto y compartir el producto	*Search* *Menu* *Shopping Cart* *Product page* *Tip a friend*	

Tabla 3.9 Composiciones de UIDPs para e-Commerce

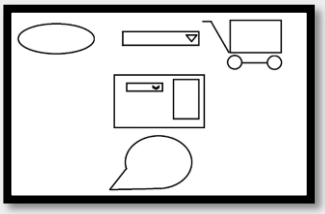
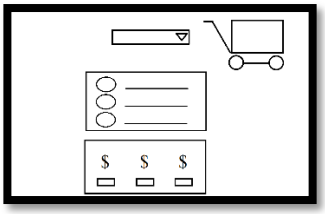
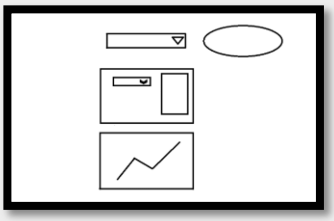
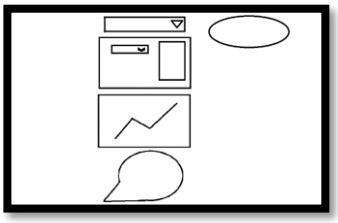
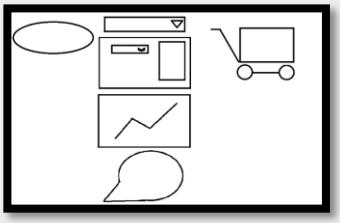
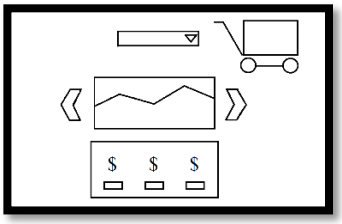
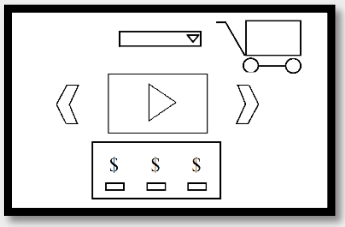
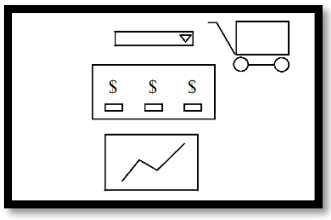
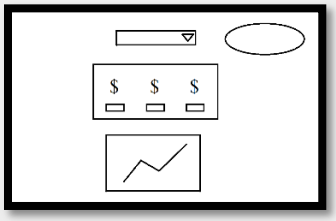
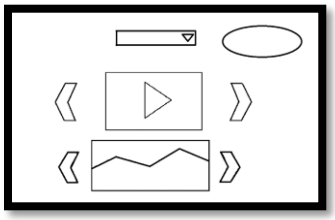
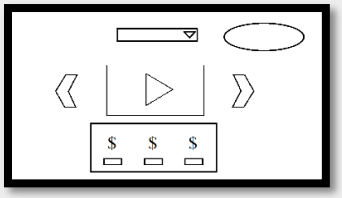
Composición	Descripción	UIDPs	Icono
Bus-men-sho-pro-com	Aplicación para comprar productos y hacer observaciones u opiniones de este	*Search* *Menu* *Shopping Cart* *Product page* *Comment*	
Bus-men-pan-pri	Aplicación para mostrar información del costo de uno o más productos	*Search* *Menu* *Panel* *Pricing Table*	
Bus-men-pro-sta	Aplicación para mostrar información del producto y las estadísticas de ventas o visitas del producto	*Search* *Menu* *Product page* *Stats*	
Bus-men-pro-sta-com	Aplicación para mostrar información del producto con la oportunidad de agregar comentarios y consultar estadísticas de ventas o visitas al producto	*Search* *Menu* *Product page* *Stats* *Comment*	
Bus-men-sho-pro-sta-com	Aplicación para mostrar información del producto, comprar, comentar y consultar estadísticas del producto visitado	*Search* *Menu* *Shopping Cart* *Product page* *Stats* *Comment*	
Men-sho-gal-pri	Aplicación que permite comprar productos, y elegir algún plan propuesto	*Menu* *Shopping Cart* *Gallery* *Pricing Table*	

Tabla 3.10 Composiciones de UIDPs para e-Commerce

Composición	Descripción	UIDPs	Icono
Men-sho-pla-pri	Aplicación que permite comprar productos, visualizar multimedia y elegir algún plan propuesto	*Menu* *Shopping Cart* *Player* *Pricing Table*	
Men-sho-pri-sta	Aplicación que permite elegir, comprar y ver sus estadísticas del producto.	*Menu* *Shopping Cart* *Pricing Table* *Stats*	
Bus-Men-pri-sta	Aplicación que permite observar información y estadísticas de cierto producto seleccionado servicio.	*Search* *Menu* *Pricing Table* *Stats*	
Bus-men-gal-pla	Aplicación acerca del producto o servicio contiene visualización multimedia de videos e imágenes	*Search* *Menu* *Gallery* *Player*	
Bus-men-pla-pri	Aplicación tipo informativo que permite elegir y ver información multimedia	*Search* *Menu* *Player* *Pricing Table*	

Prosiguiendo con el análisis se detallan las imágenes reales de las composiciones de UIDPs. En la Fig. 3.10 se muestran ejemplos reales de las composiciones propuestas en la Tabla 3.4, tales como: Bus-men-pro-com, Bus-men-pro-pla y Bus-men-pro-gal.

Bus-men-pro-com *Bus-men-pro-pla* *Bus-men-pro-gal*

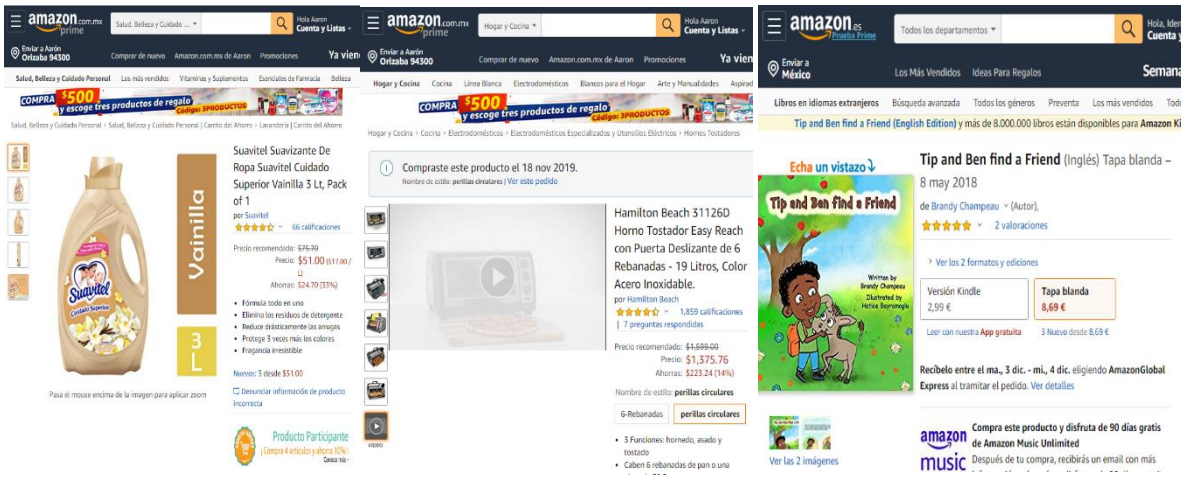


Fig. 3.10 Composiciones: Bus-men-pro-com, Bus-men-pro-pla y Bus-men-pro-gal

En la Fig 3.11 están representados las siguientes composiciones: Bus-men-pro-tip y Bus-men-sho-cou.

Bus-men-pro-tip *Bus-men-sho-cou*

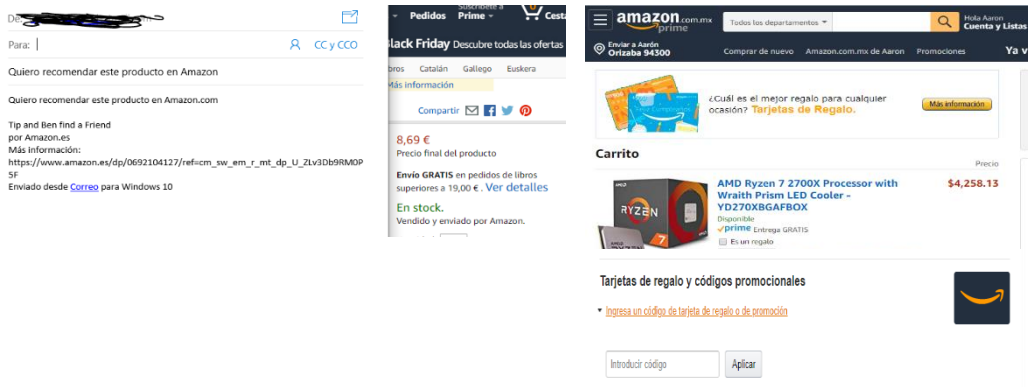


Fig. 3.11 Composiciones: Bus-men-pro-tip y Bus-men-sho-cou

Para la Fig. 3.12 las siguientes composiciones son: Bus-men-sho-pro-gal, Bus-men-sho-pro-pla y Bus-men-sho-pro-tip.

Bus-men-sho-pro-gal *Bus-men-sho-pro-pla* *Bus-men-sho-pro-tip*

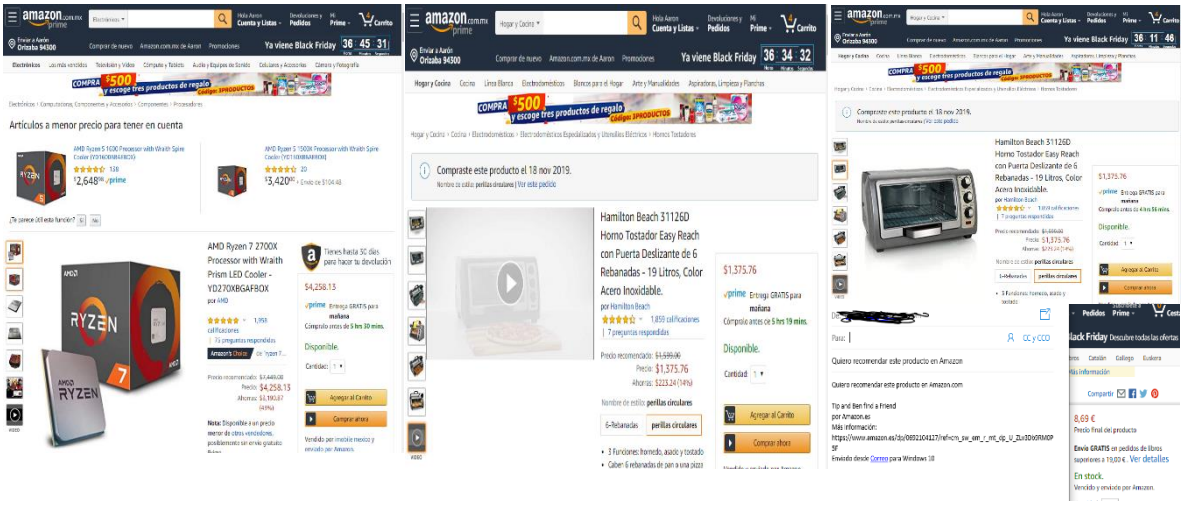


Fig. 3.12 Composiciones: Bus-men-pro-com, Bus-men-pro-pla y Bus-men-pro-gal

Para la Fig. 3.13 se representan las composiciones tales como: Bus-men-sho-pro-gal, Bus-men-sho-pro-pla y Bus-men-sho-pro-tip.

Bus-men-sho-pro-com *Bus-men-pan-pri* *Bus-men-pro-sta*

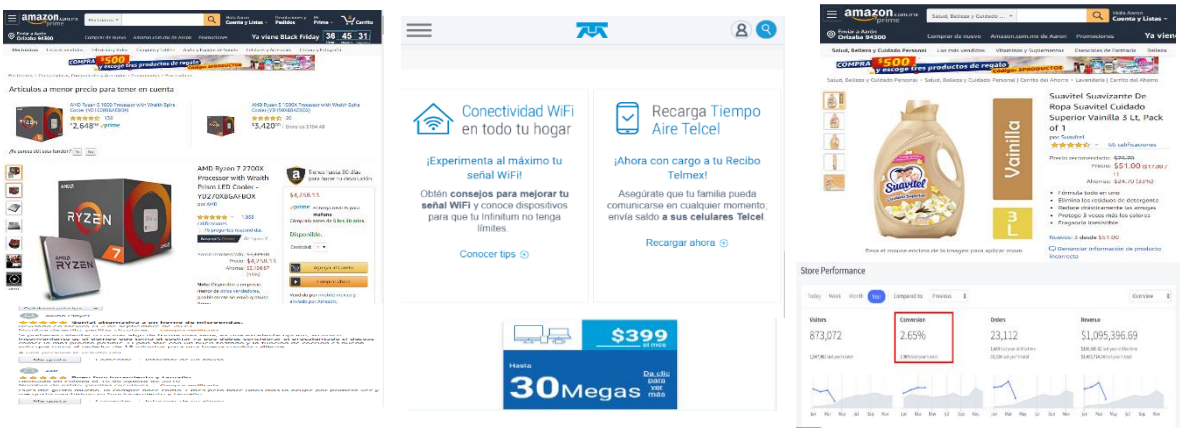


Fig. 3.13 Composiciones: Bus-men-sho-pro-com, Bus-men-pan-pri y Bus-men-pro-sta

Al continuar con la Fig. 3.14 las composiciones representadas son: Bus-men-pro-sta-com, Bus-men-sho-pro-sta-com y Men-sho-gal-pri

Bus-men-pro-sta-com *Bus-men-sho-pro-sta-com* *Men-sho-gal-pri*

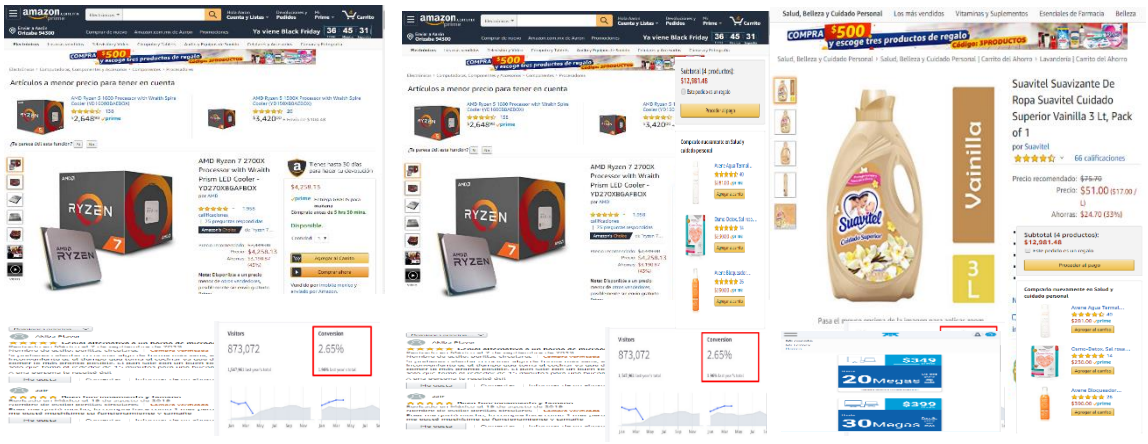


Fig. 3.14 Composiciones: Bus-men-pro-sta-com, Bus-men-pro-sta-com y Men-sho-gal-pri

La siguiente Fig. 3.15 muestra las composiciones: Men-sho-pla-pri, Men-sho-pri-sta y Bus-men-pri-sta

Men-sho-pla-pri *Men-sho-pri-sta* *Bus-men-pri-sta*

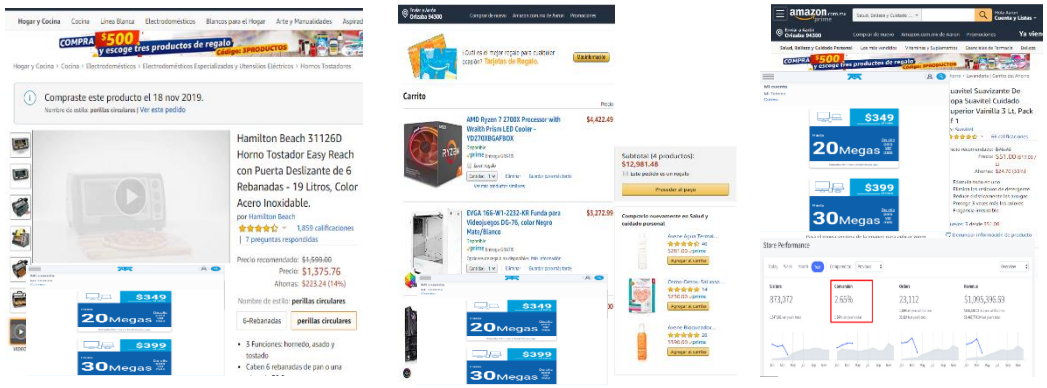


Fig. 3.15 UIDPs de composición e-Commerce: Men-sho-pla-pri, Men-sho,pri,sta y Bus-men-pri-sta

Por último, la fig. 3.16 representa los ejemplos de las composiciones: Bus-men-gal-pla y Bus-men-pla-pri.

Bus-men-gal-pla *Bus-men-pla-pri*

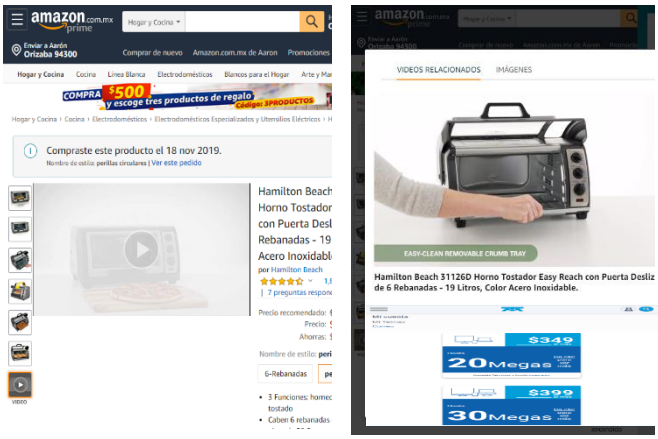


Fig. 3.16 UIDPs de composición *e-Commerce*: Bus-men-gal-pla y Bus-men-pla-pri

A continuación, en la Tabla 3.5 se describen las composiciones de UIDPs propuestas para *Social Media*.

Tabla 3.11 Composiciones de UIDPs para Social Media

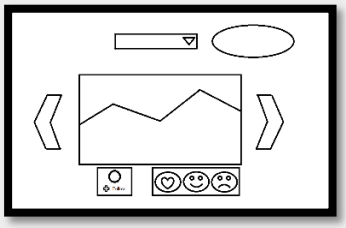
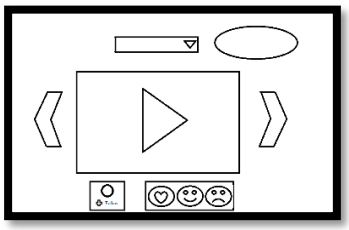
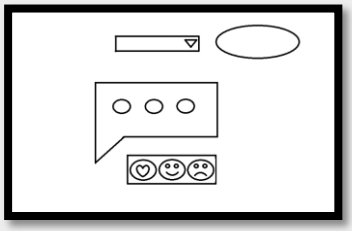
Composición	Descripción	UIDPs	Icono
Bus-men-gal-fol-rea	Aplicación que contiene las publicaciones de imágenes con capacidad de reaccionar y seguir al usuario	*Search* *Menu* *Gallery* *Follow* *Reaction*	
Bus-men-pla-fol-rea	Aplicación que contiene publicaciones de videos con capacidad de reaccionar y seguir al usuario	*Search* *Menu* *Player* *Follow* *Reaction*	
Bus-men-cha-rea	Aplicación que permite ver los chats publicados por los amigos y permite reaccionar	*Search* *Menu* *Chat* *Reaction*	

Tabla 3.12 Composiciones de UIDPs para Social Media

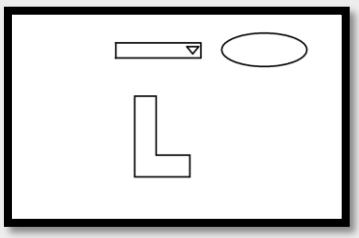
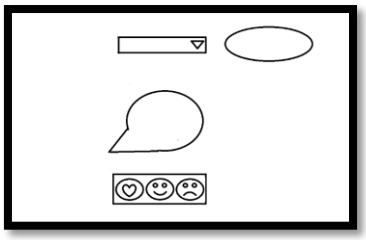
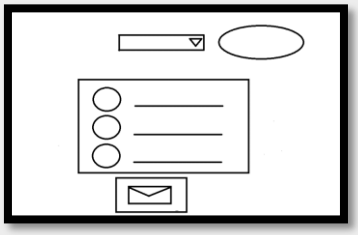
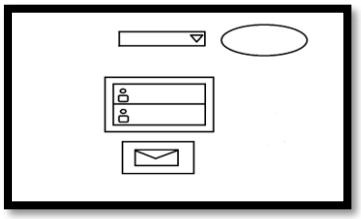
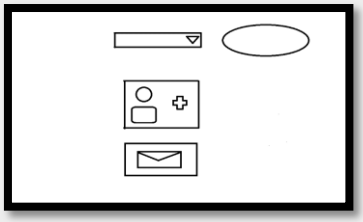
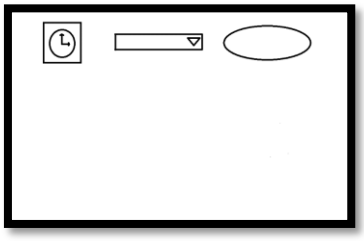
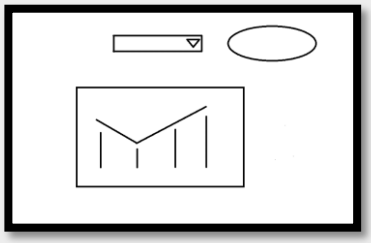
Composición	Descripción	UIDPs	Icono
Bus-men-log	Aplicación que permite iniciar sesión del usuario	*Search* *Menu* *Logging*	
Bus-men-com-rea	Aplicación que permite comentar y reaccionar en base a la publicación	*Search* *Menu* *Comment* *Reaction*	
Bus-men-pan-fin	Aplicación que permite compartir información de la publicación	*Search* *Menu* *Panel* *Friend Invite*	
Bus-men-fin-fli	Aplicación que permite invitar a más usuarios	*Search* *Menu* *Friend Invite* *Friend List*	
Bus-men-fin-fri	Aplicación para agregar e invitar a más usuarios	*Search* *Menu* *Friend Invite* *Friend*	
Bus-men-act	Aplicación que permite ver las actividades de los usuarios	*Search* *Menu* *Activity* *Stream*	

Tabla 3.13 Composiciones de UIDPs para Social Media

Composición	Descripción	UIDPs	Icono
Bus-men-ran	Aplicación que permite ver el ranking de los usuarios de una aplicación	*Search* *Menu* *Ranking*	

Una vez definidas las composiciones de UIDPs para *Social Media* se presenta la visualización que corresponde a las interfaces reales.

En la Fig. 3.17 se visualiza las siguientes composiciones *Social Media*: Bus-men-gal-pla y Bus-men-pla-fol-rea y Bus-men-cha-rea.

Bus-men-gal-pla *Bus-men-pla-fol-rea* *Bus-men-cha-rea*

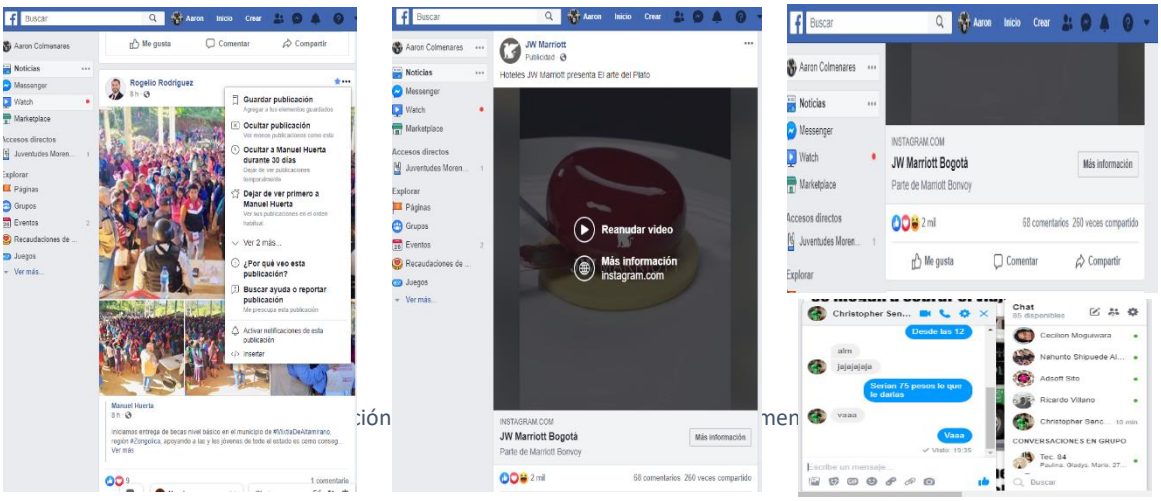


Fig. 3.17 Composiciones *Social Media*: Bus-men-log,Bus-men-com-rea y Bus-men-pan-fin

En la Fig. 3.18 se encuentran representadas las siguientes composiciones: Bus-men-log,Bus-men-com-rea y Bus-men-pan-fin.

Bus-men-log *Bus-men-com-rea* *Bus-men-pan-fin*

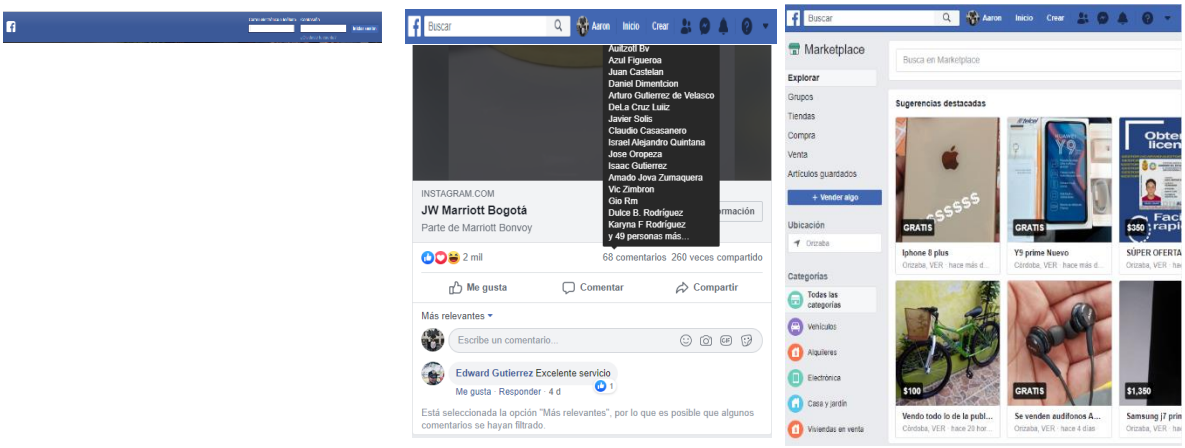


Fig. 3.18 UIDPs de composición *Social Media*: Bus-men-log,Bus-men-com-rea y Bus-men-pan-fin

Al continuar con la ejemplificación de las composiciones, en la Fig. 3.18 se observan las composiciones: Bus-men-fin-fli, Bus-men-fin-fri y Bus-men-act

Bus-men-fin-fli *Bus-men-fin-fri* *Bus-men-act*

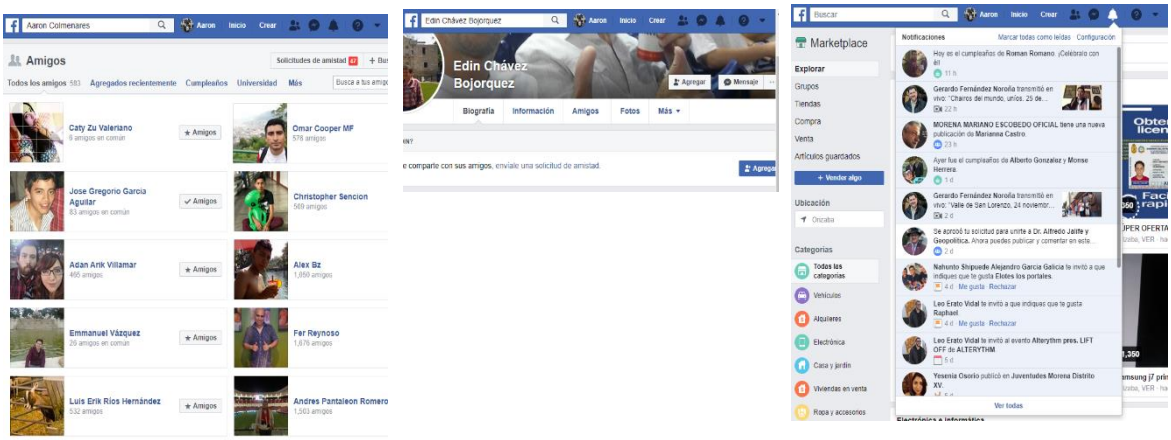
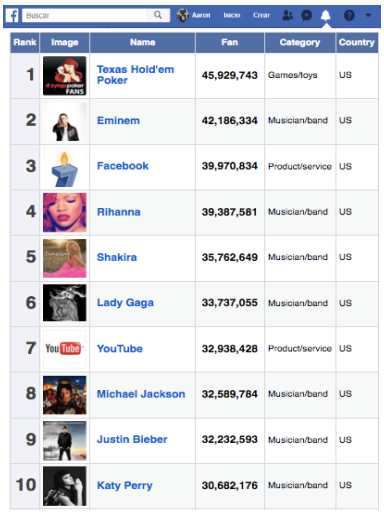


Fig. 3.19 Composiciones *Social Media*: Bus-men-fin-fli, Bus-men-fin-fri y Bus-men-act

Por último, se observa en la Fig. 3.20 la composición: Bus-men-ran.

Bus-men-ran



Rank	Image	Name	Fan	Category	Country
1		Texas Hold'em Poker	45,929,743	Games/toys	US
2		Eminem	42,186,334	Musician/band	US
3		Facebook	39,970,834	Product/service	US
4		Rihanna	39,387,581	Musician/band	US
5		Shakira	35,762,649	Musician/band	US
6		Lady Gaga	33,737,055	Musician/band	US
7		YouTube	32,938,428	Product/service	US
8		Michael Jackson	32,589,784	Musician/band	US
9		Justin Bieber	32,232,593	Musician/band	US
10		Katy Perry	30,682,176	Musician/band	US

Fig. 3.20 Composición *Social Media*: Bus-men-ran

Los patrones compuestos de e-Commerce y Social Media analizados permiten saber cuántas conexiones y relaciones existen con los UIDPs. Permite en lo general saber cómo se encuentran estructurados ciertas ventanas y tener una solución específica a la necesidad que interpreta cada UIDP compuesta.

3.3.Proceso de identificación de elementos en interfaces mediante Deep Learning

En esta sección se describen las instancias para entrenar el modelo de Deep Learning, y la descripción del modelo que especifica sus características.

3.3.1. Conjunto de imágenes

El conjunto de imágenes con que se cuentan actualmente suma un total de 47 UIDPs, los cuales se clasificaron en 4 tipos: imágenes ideales, no ideales, reales en blanco y negro y reales a color con un total de más de 18,800 instancias. Las imágenes ideales son imágenes generadas mediante cualquier herramienta de dibujo y no presentan deformaciones. Las imágenes no ideales son aquellas generadas a mano alzada, desde cualquier aplicación o herramienta de dibujo o bien a través de una tableta digitalizadora. Las imágenes reales en

blanco y negro son imágenes generadas a mano alzada en hojas de papel y digitalizadas a través de una cámara o un scanner. Finalmente, las imágenes reales a color son imágenes generadas a mano alzada, en hojas de papel de diferentes colores y utilizar bolígrafos de diferentes colores.

La resolución de todas las imágenes es de 1050 * 650 pixeles listas para probar, cada UIDP contiene un total de 400 instancias, es decir 100 por cada tipo de imagen. Cabe mencionar que de las 47 UIDPs, 32 pertenecen al dominio de Social Media y de las cuales 12 son UIDPs simples y 20 son UIDPs compuestos. Para el dominio e-Commerce se tienen 15 UIDPs, de los cuales 5 son UIDPs simples y 10 son UIDPs compuestos.

En la tabla 3.6 se observan diferentes ejemplos de UIDPs simples divididos por imágenes ideales y no ideales. Es importante mencionar que algunos UIDPs son válidos para ambos dominios: *e-Commerce* y *Social Media*.

Tabla 3.14 UIDPs simples ideales y no ideales

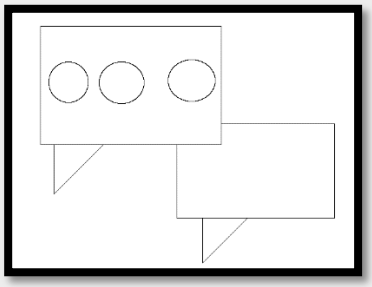
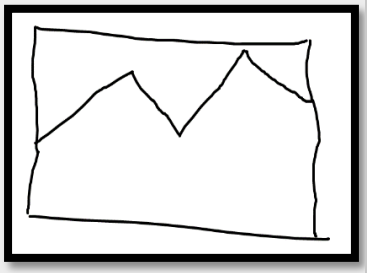
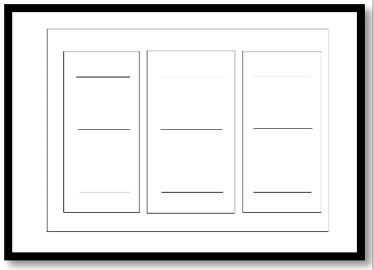
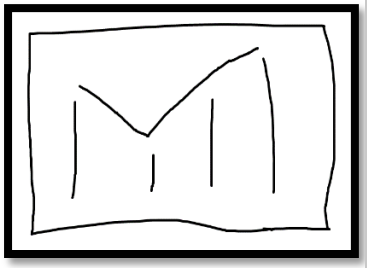
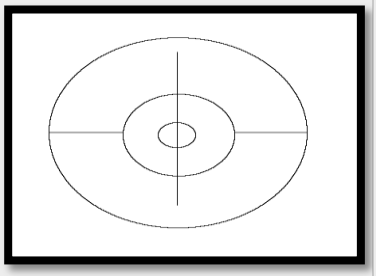
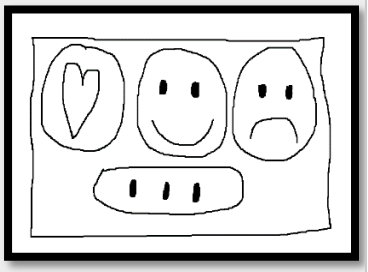
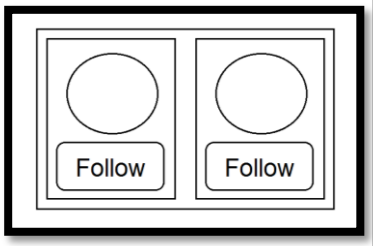
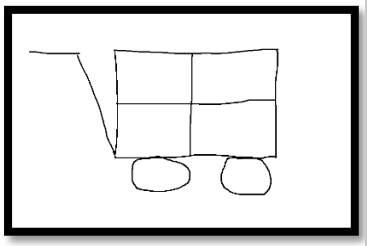
UIDP	Imágenes ideales	UIDP	Imágenes no ideales
Chat Social Media E-Commerce		***Gallery*** **Social Media E-Commerce**	
Menu-Horizontal Social Media E-Commerce		***Ranking*** **Social Media E-Commerce**	

Tabla 3.15 UIDPs simples ideales y no ideales

UIDP	Imágenes ideales	UIDP	Imágenes no ideales
Picker Social Media E-Commerce		***Reaction*** Social Media	
Follow Social Media		***Shopping cart*** e-Commerce	

En la tabla 3.7 se presenta ejemplos de composiciones divididos por imágenes reales en blanco y negro, e imágenes a color.

Tabla 3.16 Composiciones de UIDPs Reales en blanco y negro, y reales a color

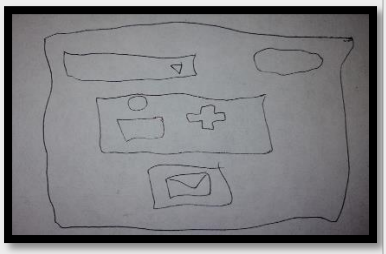
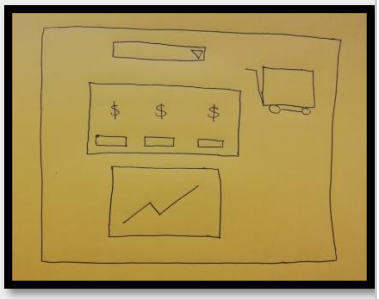
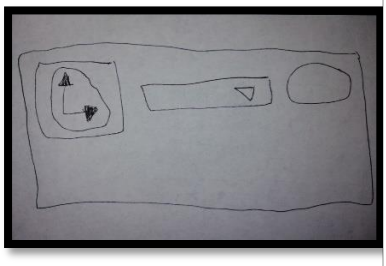
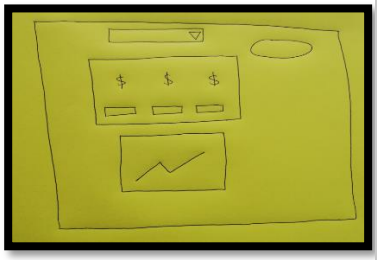
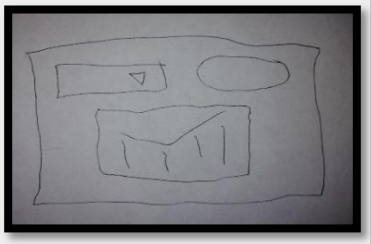
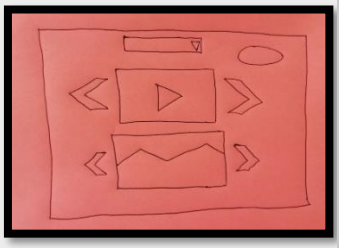
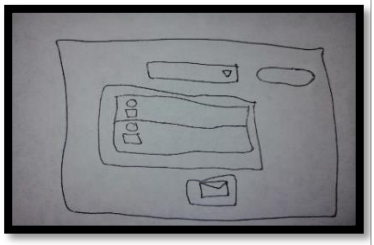
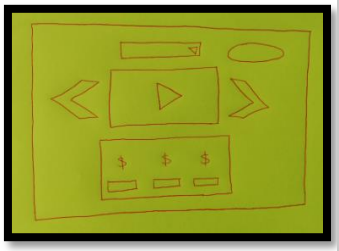
Composición	Real Blanco y Negro	Composición	Real a Color
Bus-men-fin-fri *Social Media*		***Men-sho-pri-sta*** *e-Commerce*	
Bus-men-act *Social Media*		***Bus-Men-pri-sta*** *e-Commerce*	

Tabla 3.17 Composiciones de UIDPs Reales en blanco y negro, y reales a color

Composición	Real Blanco y Negro	Composición	Real a Color
Bus-men-ran *Social Media*		***Bus-men-gal-pla*** *e-Commerce*	
Bus-men-fin-fli *Social Media*		***Bus-men-pla-pri*** *e-Commerce*	

3.4. Proceso de transformación de la imagen

Una vez creadas y definidas las instancias (imágenes) para el entrenamiento se aplicó un proceso de transformación de las instancias para las redes neuronales convolucionales. El proceso de transformación se basa en la configuración del tipo de red neuronal a utilizar. En este sentido, los estándares óptimos para la alimentación de una red neuronal incluyen valores en pixeles con un mínimo de 200 y un máximo de hasta 299 pixeles, para una óptima solución en las redes neuronales como AlexNet que requiere una resolución de la imagen de 256 * 256 pixeles, mientras que Inception_v3 requiere una resolución de la imagen de 255 * 255 pixeles para cada imagen. Desde esta perspectiva las imágenes pasan por un proceso de reorientación de la resolución para embonar en el proceso neuronal convolucional y en mejorar la calidad de precisión de las imágenes.

Para obtener un estándar específico y limpio se recomienda pasar las imágenes de color a imágenes en blanco y negro, esto ayuda completamente a disminuir el consumo computacional.

Previo al entrenamiento de la red neuronal el proceso de transformación en ambas redes neuronales consistió en: (1) se obtiene una imagen original con una dimensión total de 1050 * 650 pixeles y se redimensiona a 256 * 256 (ver Fig. 3.21-a). (2) La instancia pasa por el primer cambio que determina eliminar el ruido por medio de métodos y funciones de

procesamiento de imágenes ya implementados, los algoritmos aplicados son: *mediaBlur* que es una función que suaviza la imagen, dilatación que permite el aumento de los objetos a primer plano y por último la normalización, que se encarga de optimizar los valores altos y bajos de una matriz (ver Fig. 3.21-b). Al final, se implementa la función *Local Otsu*, encargada de optimizar un umbral del pixel para aclarar y maximizar los pixeles en dos clases, primer plano y fondo (ver Fig. 3.21-c).

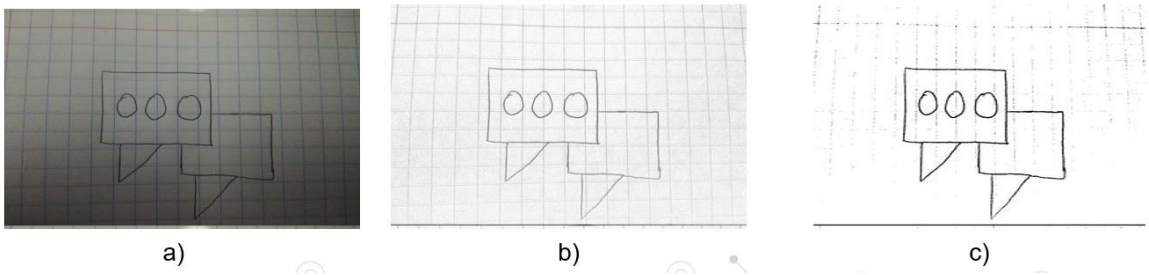


Fig. 3.21 Ejemplo de la eliminación del ruido de una imagen real a color, a) imagen real a color, b) imagen real aplicación en la eliminación de sombras, c) imagen real aplicación eliminación de ruido.

Estos métodos y funciones que se implementan por las APIs scikit-image versión 0.15.0 y opencv versión 2.4.13.1, que son bibliotecas especiales y esenciales que contienen miles de algoritmos para el procesamiento de imágenes. Una vez realizado este proceso de transformación, se continua con ejecución de los entrenamientos de las redes neuronales convolucionales.

3.4.1. Descripción del entrenamiento

Para la utilización de las redes neuronales convolucionales en este punto se hace una breve descripción específica de las redes neuronales convolucionales utilizadas para la elaboración de los resultados obtenidos.

3.4.1.1.Entrenamiento con diferentes modelos de redes neuronales

Las siguientes pruebas generadas permitieron conocer ampliamente las características que conforman las rede convolucionales tales como: los atributos, métodos y funciones que se requieren para madurar una red convolucional.

Para la prueba 1 (ver la Tabla 3.8) se contó con un total de 250 imágenes, con una configuración de 256 batch y 3 épocas. Esta prueba se realizó con un modelo clásico específico llamado *LeNet*, esta red cuenta con 6 capas convolucionales y una de salida, esta prueba generada en la API *DeepLearning4Java* permite probar con el lenguaje Java una pequeña red neuronal, la primera característica de desarrollar esta prueba permitió entender las características de cómo se instruye una red neuronal y como funciona.

Tabla 3.18 Resultados de la prueba 1

UIDPs	Precisión	Tiempo (seg)
Acordeon	0.45	21.32
Barra de etiqueta	0.45	21.32
Login	0.45	21.32
Master Detalle	0.45	21.32
Tarjeta	0.45	21.32

Para la prueba 2 (ver la Tabla 3.9) se usó la API *TensorFlow*, con un modelo que cuenta con 2 capas convolucionales y 1 capa de salida, tiene un total de 600 imágenes, con una configuración de la red neuronal convolucional con 1699 batch y 100 épocas. La prueba realizada sobre una red neuronal convolucional binominal que solamente clasifica dos imágenes, la característica importante de esta red es entender las propiedades y atributos importantes de la API *TensorFlow* en el lenguaje Python.

Tabla 3.19 Resultados de la prueba 2

UIDPs		Precisión	Tiempo (seg)
Gallery		1	30
Music Player		3.1575022e-13	45

La prueba 3 (ver la Tabla 3.10) se desarrolló con la API *TensorFlow* con un total de 800 imágenes, la importancia del entrenamiento y la prueba realizada fue utilizar el modelo anterior (prueba 2) y aumentar al doble la clasificación de los UIDPs, la red utiliza una configuración de 1401 batch y 10 épocas.

Tabla 3.20 Resultados de la prueba 3

UIDPs	Precisión	Tiempo (seg)
Gallery	0.4339219	23.407269
Menu Horizontal	0.422323233	23.407269
Menu Vertical	0.123212333232	23.407269
Panel	0.0221412	23.407269

Para la prueba 4 (ver la Tabla 3.11) se contó con un total de 800 imágenes, con una configuración de 699 batch y 10 épocas. En esta prueba se realizó una transformación en las imágenes del entrenamiento y el resultado de la transformación se implementó con la red neuronal convolucional *AlexNet* que se detalla en la siguiente sección. Cuenta con 800 imágenes, y su configuración se basa en 699 batch y 10 épocas, los resultados obtenidos se observan en la Tabla 3.13.

Tabla 3.21 Resultados de la prueba 4

UIDPs	Precisión	Tiempo (seg)
Gallery	0.99983644	8.24
Menu Horizontal	0.56006614	8.32
Menu Vertical	0.99995962	8.40
Music Player	0.9624471	8.18
Panel	0.9980167	7.98
Picker	0.8734556	7.91
Ranking	0.98580235	7.85

Como se observa, los resultados obtenidos en la prueba 4 fueron mucho más exactos, obteniendo un 99% de eficiencia en la clasificación de los UIDPs y con un tiempo de respuesta de 8 segundos en promedio.

3.4.1.2. Entrenamiento con AlexNet

AlexNet es un tipo red convolucional lineal que se implementa con la API de tensorFlow 1.12 en la versión de Anaconda 1.9.6 junto con Spyder 3.6. En la Fig. 3.22 se muestra la estructura del modelo, el cual cuenta con una configuración estandarizada de 5 capas convolucionales, además de la capa *flattening*, una capa *fully-connected* y la capa de salida. La elección este modelo se basa en la premisa de obtener una tasa de error que comprende del 26% al 15.3%, además soporta algoritmos como *max-pooling*, *dropout*, *data argumentation* y *ReLu*, que permiten obtener una estandarización de los datos, no cuenta con restricciones de tamaño o dimensiones para cada capa convolucional.

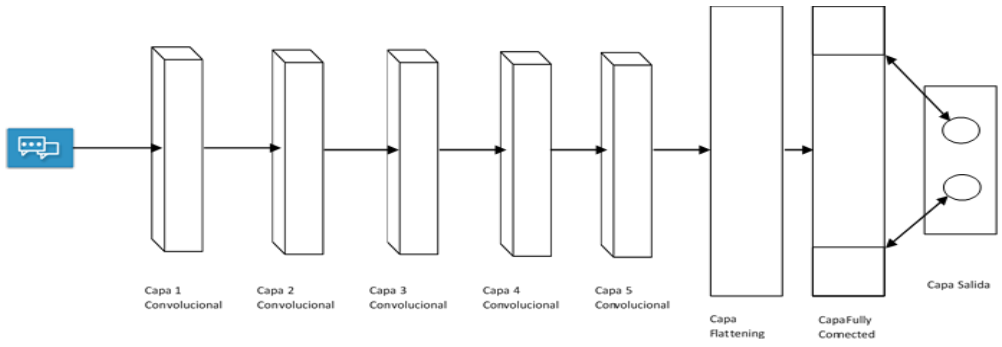


Fig. 3.22 Arquitectura de la red neuronal convolucional con 5 capas convolucionales, 1 capa flattening, 1 capa fully-con y capa de salida.

El modelo permite calcular las probabilidades de la extracción de las características del patrón resultante y usar valores gradiente, en otras palabras, se reducen los pesos generados del kernel para clasificar la imagen que corresponda a su matriz resultante.

A continuación, se abordan las pruebas realizadas para obtener el modelo final con AlexNet. Prueba 1: En la Tabla 3.12 se registran los resultados obtenidos del entrenamiento con una configuración de 1199 batch con un total de 10 épocas y un tiempo de entrenamiento de 8.7 horas, da una precisión de 88%. La importancia de hacer esta prueba es observar los resultados y validar el alcance que tiene el modelo con respecto de más de siete UIDPs que se hicieron en la sección pasada, otro punto a resaltar es la implicación del proceso de transformación que ayuda a mejorar y concentrar las imágenes necesarias para entrenar.

Con nueve patrones y con 200 instancias el modelo AlexNet cumple con los resultados de precisión, derivado a que se tomarán los cuatro primeros resultados, en la tabla los valores subrayados en rojo denotan los resultados obtenidos, donde 1 se toma como un correcto acercamiento de los cuatro primeros lugares cercanos al 100% y 0 donde no se encuentra en los cuatro primeros lugares de clasificación al 100%.

Tabla 3.22 Tabla de entrenamiento para AlexNet con 9 patrones y 200 instancias c/u

	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Chat	1	0.000162506	0.19352973	0.000473471	0.068136394	0.002334441	1.16E-05	0.03520408	0.003349891
Gallery	0.013060846	**1**	0.001202669	4.30E-05	0.007741266	0.001050286	5.16E-05	0.006903645	0.010544525
Picker	0.000459286	8.85E-05	1	4.91E-05	0.10553073	0.000117258	0.003349891	0.002373413	0.000523853
Panel	0.00055751	3.29E-05	0.005648395	1	0.023106337	0.301504659	0.000206908	0.000674044	0.000447564
Comment	0.000514463	0.001669927	0.14482954	0.000126492	1	0.450005678	4.71E-06	0.26613018	0.000487
Menu horizontal	0.002287021	0.000925624	0.001918043	0.001022949	0.012773899	0	0.000415436	0.000210459	0.002145337
Menu vertical	0.0000544	0.0004578	0.000532496	0.000565931	0.001238559	0.24860461	1		0.001320293
Music player	0.000209294	0.000626944	0.034880344	2.61E-05	0.018747807	0.005845337	2.07E-05	1	0.001445827
Ranking	0.000182633	0.002265977	.0044776	.001279	0.001068667	0.003140203	4.57E-05	0.001355291	1

Prueba 2: En esta segunda prueba se utilizó un conjunto de 9 patrones con 400 imágenes cada uno, por lo tanto, un total de 3600 imágenes el doble de la prueba anterior, esto fue gracias a una nueva versión del proceso de transformación de la imagen. En la Tabla 3.13 se registran los resultados del entrenamiento que se obtuvo con una configuración de 1199 batch con un total de 10 épocas con un tiempo de entrenamiento de 11.6 horas, da una precisión del 100%. La importancia de hacer esta prueba fue ver el comportamiento de usar la misma cantidad de patrones de la prueba 1.

Tabla 3.23 Tabla de entrenamiento para AlexNet con 9 patrones y 400 instancias c/u

	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Chat	Gallery	Picker
Chat	1	0.24458238	0.09221611	0.009311597	0.077810146	0.19702998	0.012670485	0.000776279	0.007370127
Gallery	0.038561948	1	0.050907224	0.002415942	0.005094664	0.041085303	0.006743959	3.76E-05	0.001321756
Picker	0.018224	0.000546648	4	0.006819897	0.045885 6	0.012618925	0.000145164	0.000177965	0.005822974
Panel	0.006833378	0.027818032	0.13985293	1	0.272508 68	0.061365534	0.000859995	0.000224905	0.009056208
Comment	0.005135891	0.014125279	0.05282312	0.051087 1	1	0.019245375	0.000551739	1.04E-05	0.006095901
Menu horizontal	0.19797277	0.06933659	0.00417035	0.001078468	0.012612 53	1	0.002419511	0.000106045	0.001091085
Menu vertical	0.006887948	0.05686648	0.055002443	0.003285562	0.005871234	0.06844314	1	0.000152222	0.16353118
Music player	0.005039374	0.002415473	0.10943875	0.000455572	0.000866247	0.010981618	0.018856758	1	0.001316326
Ranking	0.000763028	0.002366882	0.004325727	0.000673173	0.001089388	0.004886695	0.000449571	7.39E-05	1

Prueba 3: para esta prueba se utilizaron 17 patrones con 400 imágenes cada uno, es decir un total de 6800 instancias. En la Tabla 3.14 se registran los resultados obtenidos del entrenamiento con una configuración de 1499 lotes con un total de 10 épocas con un tiempo de entrenamiento de 19.4 horas, da una precisión de 52%. El motivo de esta prueba es aumentar 10 UIDPs, para saber el comportamiento de la red neuronal, y como se observa la precisión cae considerablemente. A pesar que se cuentan con más imágenes, la cantidad de UIDPs que también incrementa, hace que la precisión no sea óptima.

Tabla 3.24 Tabla de entrenamiento para AlexNet con 17 patrones y 400 instancias c/u

	Composición 1	Composición 2	Composición 3	Composición 4	Composición 5	Composición 6	Chat
Composición 1	1	0.24458238	0.09221611	0.009311597	0.077810146	0.19702998	0.012670485
Composición 2	0.038561948	0	0.050907224	0.002415942	0.005094664	0.041085303	0.006743959
Composición 3	0.018224	0.000546648	0	0.006819897	0.0458856	0.012618925	0.000145164
Composición 4	0.006833378	0.027818032	0.13985293	1	0.27250868	0.061365534	0.000859995
Composición 5	0.005135891	0.014125279	0.05282312	0.0510871	0	0.019245375	0.000551739
Composición 6	0.19797277	0.06933659	0.00417035	0.001078468	0.01261253	0	0.002419511
Chat	0.006887948	0.05686648	0.055002443	0.003285562	0.005871234	0.06844314	1

Tabla 3.25 Tabla de entrenamiento para AlexNet con 17 patrones y 400 instancias c/u

	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Gallery	1	0.0013163 26	6.29E-05	0.0091310 86	0.00077163	6.96E-05	0.00872514	0.0129698 24
Picker	7.39E-05	1	5.29E-05	0.1105407 5	5.36E-05	.000547877	0.00272435	0.0004363 17
Panel	2.20E-05	0.0040388 93	1	0.0210363 36	0.000931349	0.000236409	0.00057979 2	0.0004096 68
Comment	0.0013741 14	0.1302709 6	0.0001516 63	1	0.027818032	4.98E-06	0.24293001	0.0001934
Menu horizontal	0.0006370 18	0.0012962 12	0.0010562 42	0.0096009 7	0	0.00058558	0.00017455 9	0.0020066 37
Menu vertical		0.0004090 19	0.0009045 08	0.0011317 09	0.16476302	1		0.0012816 88
Music player	0.0006151 63	0.0341845 2	3.42E-05	0.0199783 78	0.000356791	2.25E-05	1	0.0013321 99
Ranking	0.0024393 73	.0024757	3.20E-05	0.0012595 05	0.002316048	6.23E-05	0.00145763 3	1

Prueba 4: En esta prueba se utilizaron 27 patrones con 400 imágenes cada uno, por lo tanto, se cuenta con un total de 10,800 imágenes. En la Tabla 3.15 se registraron los resultados obtenidos del entrenamiento con una configuración de 2993 batch, un total de 10 épocas y un tiempo de entrenamiento de 37.3 horas, da una precisión de 43.3%. Se observa que la precisión sigue disminuyendo mientras más UIDPs se clasifiquen, por lo tanto, AlexNet en esta última prueba no logra un resultado satisfactorio para reconocer los patrones

Tabla 3.26 Tabla de entrenamiento para AlexNet con 27 patrones (1)

	Composición 1-s	Composición 2-s	Composición 3-s	Composición 4-s	Composición 5-s	Composición 6-s	Composición 7-s	Composición 8-s	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Composición 1-s	1	0.0923282	0.0690347	0.0039042	0.0242505	0.0312753	0.0141203	0.0068902	0.0112759	0.0007363	0.0032910	0.0003366	0.0005541	0.1339235	0.0003173	9.42E-05	0.0005759
Composición 2-s	0.0177782	0	0.0049083	0.0028915	0.0039513	0.0150013	0.0924605	0.0006503	0.0008191	0.0002293	0.0017565	2.32E-05	0.0119035	0.0102246	5.36E-05	0.0001302	0.0028794
Composición 3-s	0.0133765	0.0010219	0	0.0068792	0.0291133	0.0047169	0.0012931	0.0065077	0.0000370	0.0000489	0.0037345	3.41E-05	0.0002174	0.0199633	8.12E-05	0.0001993	0.0024553
Composición 4-s	0.0042076	0.0188518	0.0166366	0	0.0545076	0.0211611	0.0484729	0.0033909	0.0000950	0.0004394	0.0079482	3.99E-05	0.0135429	0.0015305	2.97E-06	0.0002544	0.0006160

Tabla 3.27 Tabla de entrenamiento para AlexNet con 27 patrones (1)

	Composición 1-s	Composición 2-s	Composición 3-s	Composición 4-s	Composición 5-s	Composición 6-s	Composición 7-s	Composición 8-s	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Composición 5-s	0.00331273 5	0.00878274 4	0.00209723 3	0.02001338 8	1	0.00669 2737	0.07712532 6	0.00303118 8	0.0005671 43	2.70E-05	0.0054506 26	1.89E-05	0.0036608 85	0.0002685 17	2.35E-06	9.87E-05	0.0002407 19
Composición 6-s	0.05577554	0.04947731 6	0.00346896 5	0.00119103 2	0.0074769 75	0	0.04757481	0.00036018 7	0.0024306 61	0.0001664 62	0.0015335 04	2.03E-05	0.0006706 8	0.0009202 08	1.31E-05	0.0004163 06	0.0017747 49
Composición 7-s	0.00230558 9	0.01884681	0.00036990 9	0.00016281 5	0.0003546 92	0.031546798	1	0.00024924 2	0.0007066 19	3.90E-06	0.0029311 1	5.82E-07	0.0012013 65	0.0016260 83	0.0008101 461	1.71E-05	0.0002956 2
Composición 8-s	0.00736999 3	0.00329495 1	0.07192188	0.01574792 9	0.0819610 5	0.030801853	0.00116141 5	1	0.0004671 23	1.84E-05	7.97E-05	0.0000546 515	0.0001959 89	0.1498 4247	7.72E-05	2.05E-05	0.0006317 5

Tabla 3.28 Tabla de entrenamiento para AlexNet con 27 patrones (1)

	Composición 1-s	Composición 2-s	Composición 3-s	Composición 4-s	Composición 5-s	Composición 6-s	Composición 7-s	Composición 8-s	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Chat	0.00436719	0.04129565	0.00661078 2	0.00188419 1	0.00 1434 354	0.02 9469 792	0.00198691 2	5.42E -05	1	0.0 00 43 79 33	0.0 583 708 85	0.0 00 16 34 63	0.03 0441 832	0.001 02559 6	1.11 E-05	0.01 935 448 1	0.00 485 952
Gallery	0.00171210 2	0.00143546 9	0.01305518 2	0.00031032 6	0.00 0238 446	0.00 4192 975	7.75E -05	1.27E -06	0.0 109 586 17	1	0.0 004 388 67	3.1 0E -05	0.00 2743 303	0.000 43470 9	5.87 E-05	0.00 462 401 4	0.01 764 701 9
Picker	0.000 80096 3	0.002 74907	0.000 24815 5	0.000 80885 5	0.00 0574 844	0.00 3728 546	0.002 07176 3	1.38E -05	0.0 005 981 84	0.0 00 21 54 22	0	2.9 8E -05	0.05 8692 82	5.17E -05	1.42 E-06	0.00 263 075 8	0.00 049 505 8
Panel	0.000 42930 4	0.001 05682 3	0.008 00842 5	0.002 60976 7	0.00 2348 591	0.00 7961 731	0.000 23127 7	5.08E -05	0.0 008 814 34	7.6 1E- 05	0.0 025 118 41	1	0.01 0644 656	0.001 71456 2	0.00 0717 219	0.00 046 736 4	0.00 053 758 5
Comment	0.000 42355 8	0.001 51988 1	0.000 95145 6	0.000 34116	0.00 0332 541	0.00 3164 45	0.000 68312	4.10E -06	0.0 008 291 79	0.0 04 33 99 3	0.1 066 047 2	9.5 7E -05	0	4.35E -05	6.26 E-06	0.25 125 784	0.00 028 112 2

Tabla 3.29 Tabla de entrenamiento para AlexNet con 27 patrones (1)

	Composición 1-s	Composición 2-s	Composición 3-s	Composición 4-s	Composición 5-s	Composición 6-s	Composición 7-s	Composición 8-s	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Menu horizontal	0.000334235	0.000194345	0.01723277	0.000729235	0.005799484	0.003524236	7.69E-05	0.000696196	0.001549927	0.000078 6086	0.000451758	0.000057 1059	0.002897626	0	0.001257752	8.30E-05	0.001838192
Menu vertical	0.000342366	0.000220479	0.005334792	0.001194973	0.002093856	0.001196733	9.66E-05	4.50E-05	0.000115129	2.51E-05	0.000266223	0.000162 2568	0.000739078	0.17851332	1	4.13E-05	0.001782242
Music player	0.001793056	0.000898667	0.002562027	0.001457402	0.000845035	0.003100361	0.00043427	1.07E-05	0.000420651	0.000203 7522	0.000267 3712	2.69E-05	0.013851255	0.0002516 32	2.24E-05	0	0.001600678
Ranking	0.000133632	7.31E-05	0.000934398	6.46E-05	5.50E-05	0.000474 62	4.68E-04	1.19E-06	0.000603329	0.000872 0716	0.000258219	2.80E-05	0.000811252	0.0022930 31	9.50E-05	0.001426913	1

Tabla 3.30 Tabla de entrenamiento para AlexNet con 27 patrones (1)

	Composición 1-s	Composición 2-s	Composición 3-s	Composición 4-s	Composición 5-s	Composición 6-s	Composición 7-s	Composición 8-s	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Composición 1-e	0.10881881	0.07971123 6	0.06704838	0.00370420 6	0.00171124 1	0.15200008	0.01712563 6	5.10E-05	0.00911392 2	0.00436325 9	0.00437737 7	8.18E-06	0.00030396 6	0.01674330 2	0.00011894 1	0.00011738 5	0.00019642 5
Composición 2-e	0.06883622	0.02732702 5	0.01010791 8	0.00428653 1	0.00041699 6	0.02253540 8	0.00701835 4	3.92E-05	0.00272021	0.00166476 4	0.00146421 7	4.64E-05	0.01079172 6	0.01185047 7	0.00039244 1	5.20E-05	0.00063687 5
Composición 3-e	0.00651681 1	0.00051992 4	0.05142821	0.00338299 1	0.01317794 7	0.00813607 2	0.00062055 4	0.00318605 8	0.00023326 2	0.00051094 4	0.00264193	1.79E-05	0.00141259 2	0.01718010 4	4.86E-05	0.00014048	0.00297176 5
Composición 4-e	0.00424569 9	0.01458559 7	0.01906332 2	0.17593181	0.04869049	0.01789418 8	0.04764857	0.00333644 1	0.00080298 3	0.00034261 2	0.00901257 3	3.78E-05	0.01630851 4	0.00174916 3	3.36E-06	0.00027155 9	0.00050779 8

Tabla 3.31 Tabla de entrenamiento para AlexNet con 27 patrones (1)

	Composición 1-s	Composición 2-s	Composición 3-s	Composición 4-s	Composición 5-s	Composición 6-s	Composición 7-s	Composición 8-s	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Composición 5-e	0.00238524 5	0.00807921 5	0.00191578 6	0.01667592 5	0.07143009	0.00643720 7	0.05304356 7	0.00254646	0.00052150 9	2.70E-05	0.00486443	1.52E-05	0.00027750 92	0.00020042 7	1.83E-06	8.48E-05	0.00018738
Composición 6-e	0.09821650 4	0.08156671 4	0.00584473 5	0.00179352	0.00910125 5	0.11631474	0.06416419	0.00029998 3	0.00277656	0.00034260 6	0.00177082	1.25E-05	0.00071038 7	0.00115321 3	1.90E-05	0.00054681	0.00212367 4
Composición 7-e	0.01308052 3	0.01081210 9	0.00404028 9	0.00726324 2	0.00262687 4	0.01838058	0.00086151 4	0.00012692	0.00085350 3	0.00062449 6	0.01425195 9	0.05599 38	0.00770779 1	0.00577308 5	.0023124 5	0.00025223 9	0.00020537 6
Composición 8-e	0.01608332 6	0.03968190 4	0.00096411 5	0.00317336 5	0.00359487 1	0.01270717 2	0.01561844 4	0.00026641 7	0.00028170 4	8.20E-05	0.01222856	1.02E-05	0.00174287 6	0.00077055	0.00031449	0.00014434 2	0.00087668 5

Tabla 3.32 Tabla de entrenamiento para AlexNet con 27 patrones (1)

	Composición 1-s	Composición 2-s	Composición 3-s	Composición 4-s	Composición 5-s	Composición 6-s	Composición 7-s	Composición 8-s	Chat	Gallery	Picker	Panel	Comment	Menu horizontal	Menu vertical	Music player	Ranking
Coupon	0.02172192 7	0.04893888 5	0.09456854	0.16131629	0.02447903 9	0.08124992	0.01355337	0.00010599 8	0.0132801 69	0.0003085 43	0.0072934 55	6.00E-05	0.0824808 8	0.0008589 83	9.73E-05	0.0030989 34	0.0019327 1
Pricing Table	0.0065899 33	0.0025004 19	0.0796556	0.0756525	0.1068187 95	0.0262871 12	0.0011098 29	0.0019108 42	0.0008465 68	0.0001767 927	0.0014800 65	3.39E-05	0.0047967 93	0.0140554 1	0.0003397 9	0.0003056 77	0.0029790 63
ProductPage	0.0345176 35	0.0244521 44	0.0952573	0.0149551 3	0.0082796 16	0.0821177 4	0.0052271 18	0.0002321 52	0.0056359 228	0.0003165 253	0.0030672 28	9.05E-05	0.0095026 88	0.1614817	0.0023056 98	0.0004361 18	0.0266692 49
ShoppingCart	0.0010084 71	0.0046645 33	0.0165302 94	0.0056766 79	0.0129758 72	0.0491012 3	0.0007521 71	0.0003647 56	0.0071605 89	0.0020606 049	0.0061175 18	0.0038017 67	0.0058026 24	0.0015167 02	6.12E-05	0.0074086 44	0.0028744 04
Tip a Tip	0.0201673 77	0.0219508 19	0.0042635 05	0.0012919 02	0.0014617 74	0.0224621 84	0.0055235 94	8.95E-06	0.0587244 03	0.0071407 326	0.2885469 2	5.32E-05	0.3141359	0.0019797 04	3.98E-05	0.1359680 3	0.0032821 17

Tabla 3.15 Tabla de entrenamiento de AlexNet con 27 patrones (2)

	Composición 1-e	Composición 2-e	Composición 3-e	Composición 4-e	Composición 5-e	Composición 6-e	Composición 7-e	Composición 8-e	Coupon	Pricing Table	Product Page	Shopping Cart	Tip a Tip
Composición 1-s	0.0012 22212	0.0084 98555	0.0002 15067	0.0028 36069	0.0002 29817	0.0488 49013	0.2125 0871	0.0023 97288	0.00 032 37	5.25E -05	0.011 57643 7	0.0013 70623	0.000 3645 82
Composición 2-s	0.0084 54538	0.0071 16523	0.0028 80617	0.0172 3857	0.0060 27459	0.0047 00975	0.0019 59268	0.0007 81094	0.00 946 63	0.000 32591 4	0.003 75869	0.0002 70088	0.000 3059 73
Composición 3-s	0.0035 02943	0.0241 0948	0.0025 1941	0.0128 42772	0.0046 44138	0.0009 64578	0.0132 75475	0.0073 89479	0.00 027 95	0.003 51080 9	0.000 33671 1	0.0002 24094	0.000 2649 11
Composición 4-s	0.0067 86767	0.0013 24369	0.0102 80787	0.0046 37126	0.0180 96015	0.0001 64309	0.0050 26802	0.0035 97574	0.21 091 72	0.003 74797 1	2.33E -05	0.0005 43934	4.69E -05
Composición 5-s	0.0089 20413	0.0021 99738	0.0463 43703	0.0061 63945	0.0133 66538	0.0001 43894	0.0029 65571	0.0039 7718	0.18 474 62	0.001 23871 5	0.011 57765 7	6.07E- 05	9.80E -06
Composición 6-s	0.0011 11792	0.0035 92488	0.0118 72754	0.0008 62901	0.0011 86524	0.0037 49183	0.0196 87437	0.0002 19391	0.00 074 28	6.69E -05	0.000 16704 3	0.0013 41972	1.14E -05
Composición 7-s	0.0002 63427	0.0003 32229	0.0219 78298	0.0030 96926	0.0005 47146	0.0489 09252	0.0003 799	1.90E- 05	0.02 954 78	9.97E -05	1.10E -05	0.0002 52415	9.471 82-07
Composición 8-s	0.0012 89974	0.0087 72186	0.0002 4078	0.0004 73908	0.0003 17529	0.0011 3867	0.0010 55921	0.0003 83838	0.00 046 92	0.000 19493 3	0.002 55516 3	0.0002 69062	8.86E -06
Chat	0.1686 8266	0.0693 5984	0.1240 7381	0.0199 92307	0.0178 09797	0.0022 36657	0.0307 5584	0.0006 29054	0.00 824 97	0.004 79131 3	0.003 62118 2	0.0012 5251	0.006 1703 76

Tabla 3.15 Tabla de entrenamiento de AlexNet con 27 patrones (2)

	Composición 1-e	Composición 2-e	Composición 3-e	Composición 4-e	Composición 5-e	Composición 6-e	Composición 7-e	Composición 8-e	Coupon	Pricing Table	Product Page	Shopping Cart	Tip a Tip
Gallery	0.031075047	0.006467854	0.003559057	0.006672331	0.00590452	0.005120111	0.000348792	0.000105513	9.50E-05	0.0090024	0.17529613	0.000420665	0.1969991 9
Picker	0.002552322	0.001531312	0.018846426	0.000705575	0.005027881	0.000101329	0.00066649	8.02E-05	0.0417174	0.0007441 31	1.01E-05	0.000490869	8.93E-05
Panel	0.000562693	0.004174547	0.000527666	0.001071659	0.000839345	0.007580689	0.01061434	0.000166724	0.0005838	0.0014332 17	0.0013971 39	0.001098 01	4.67E-05
Comment	0.004203074	0.005077907	0.023654275	0.000948 83	0.010136013	0.001415114	0.002370329	0.00013232	0.0079052	0.0002291 72	7.76E-05	0.002243017	0.0025404 76
Menu horizontal	0.000626785	0.009095392	0.000199 37	0.011735376	0.003338289	0.00745432	0.002424878	0.005160174	0.0001423	0.0420526 38	0.005427169	0.001212 45	0.0002363 73
Menu vertical	0.000143034	0.004443634	5.02E-05	0.002789 5	0.000230323	0.025126405	0.020920135	0.004584726	9.37E-05	0.0004284 63	0.0024899994	0.000136 83	1.89E-05
Music player	0.002351448	0.003399683	0.001516045	0.001063561	0.00454646	0.000568046	0.002142363	4.03E-05	4.232E-05	0.0004695 17	0.0003099 79	0.000900845	0.0010711 2

Tabla 3.15 Tabla de entrenamiento de AlexNet con 27 patrones (2)

	Composición 1-e	Composición 2-e	Composición 3-e	Composición 4-e	Composición 5-e	Composición 6-e	Composición 7-e	Composición 8-e	Coupon	Pricing Table	Product Page	Shopping Cart	Tip a Tip
Ranking	0.00066 0585	0.00233 7842	0.00053 3077	0.00119 4976	0.00028 9639	0.00064 6346	0.00019 3118	1.14E-05	0.00 0470 1	0.0083 86015	0.0050 01901	0.0003 70232	0.001 8218 08
Composición 1-e	0	0.08202 814	0.01079 4097	0.05539 7544	0.01186 005	0.05525 4135	0.01737 3513	0.0007 51836	0.00 0119 9	0.0015 65359	0.0283 98156	0.0004 93001	0.003 9231 69
Composición 2-e	0.01120 3741	0	0.00438 4674	0.01322 0945	0.02722 5675	0.00793 0527	0.00593 4994	0.0001 47061	0.00 3509 4	0.0026 74006	0.0298 6511	0.0001 114	0.001 0461 89
Composición 3-e	0.00193 9175	0.01278 5941	0	0.00860 987	0.00300 8344	0.00054 6343	0.00627 8422	0.0039 43317	0.00 0144 2	0.0029 84697	0.0002 6868	0.0001 93247	0.000 1826 18
Composición 4-e	0.00617 0199	0.00146 8711	0.00977 3295	0	0.01848 7513	0.00017 3705	0.00589 0452	0.0036 60766	0.10 0617 7	0.0039 63298	2.31E-05	0.0005 34539	5.05E -05

Tabla 3.15 Tabla de entrenamiento de AlexNet con 27 patrones (2)

	Composición 1-e	Composición 2-e	Composición 3-e	Composición 4-e	Composición 5-e	Composición 6-e	Composición 7-e	Composición 8-e	Coupon	Pricing Table	Product Page	Shopping Cart	Tip a Tip
Composición 5-e	0.006189026	0.001547917	0.027481943	0.00444096	0	9.05E-05	0.001565817	0.003014107	0.142279 6	0.001210387	1.71E-06	0.001424923	7.06E-06
Composición 6-e	0.001820942	0.005773742	0.019176602	0.001489398	0.001695659	1	0.028652415	0.000030191	0.000994 4	0.000107239	0.000 242739	0.001273446	2.00E-05
Composición 7-e	0.002307526	0.011283241	0.001204547	0.039878763	0.005167937	0.2687211	0	0.000495149	0.000 70883	0.000284401	0.012 815164	0.003300189	0.000219851
Composición 8-e	0.034981206	0.020216295	0.021932047	0.2637765	0.151382 8	0.002649222	0.031448323	1	0.001 25335	0.001240397	1.60E-05	0.000161003	5.43E-05
Coupon	0.111417 71	0.062423 81	0.181117 74	0.016447896	0.164633 14	0.003787947	0.001252 88	0.000387939	0	0.015185501	0.002 384813	0.000577009	0.001039606

Tabla 3.15 Tabla de entrenamiento de AlexNet con 27 patrones (2)

	Composición 1-e	Composición 2-e	Composición 3-e	Composición 4-e	Composición 5-e	Composición 6-e	Composición 7-e	Composición 8-e	Coupon	Pricing Table	Product Page	Shopping Cart	Tip a Tip
Pricing Table	0.013906231	0.04128359	0.00327 6981	0.10608325	0.00981 0957	0.00029 8047	0.00519 3083	0.19870315	0.00 10569	0	6.36E-05	0.00292 8054	6.34 E-05
Product Page	0.03100 0808	0.08600 34	0.00494 0166	0.00444 4527	0.08930 9745	0.00825 8119	0.00954 0436	0.00287 7597	0.00 08145	0.04 069 454 2	0	0.00260 5956	0.00 2674 726
Shopping Cart	0.00341 6761	0.00441 7317	0.00624 7829	0.00126 6395	0.00299 8278	0.00298 3385	0.21376 601	0.00283 8019	0.00 0520 1	0.00 184 918 8	0.001 19871 3	1	0.00 0711 557
Tip a Tip	0.08044 1624	0.04718 3674	0.00129 5769	0.08085 692	0.06581 27	0.04638 522	0.03645 029	0.00185 8326	0.01 0740 1	0.00 719 679 4	0.016 11805 5	0.00483 4933	1

Debido a la baja precisión que se obtienen como resultado en las últimas pruebas de AlexNet se diseñó un nuevo modelo llamado *Inception* con la versión 3 que reducen principalmente el margen de error, esto permite mejorar los resultados derivados en AlexNet.

3.4.1.3.Entrenamiento con Inception_v3

Inception_v3, red neuronal convolucional modular se implementó con la API TensorFlow 1.14, Keras 2.2.5 y Anaconda 1.9.6 junto con la versión de Spyder 3.6. La red es ganadora del reto ImageNet 2014, esta red proporciona una mejoría con respecto a AlexNet. Además, la red cuenta con 22 capas profundas, esto es 8 capas más que AlexNet. Tiene mejor eficiencia con respecto al poder computacional, casi dos veces menor al proceso en AlexNet, una mayor precisión, menor uso de la memoria, contiene menos parámetros, es muy usada para dispositivos móviles, además de ser modular, permite escoger automáticamente el kernel convolucional y mejorar el poder computacional. La red usa filtros de reducción de dimensiones de capas convolucionales 1*1 para convertir automáticamente una imagen RGB de 256*256*3 a 256*256*1, además viene acompañado de capas convolucionales de 3*3, 5*5 y de 7*7 dependiendo la elección del kernel convolucional, al final se aplica Max-Pooling para mejorar los filtros de los algoritmos (ver Fig 3.23).

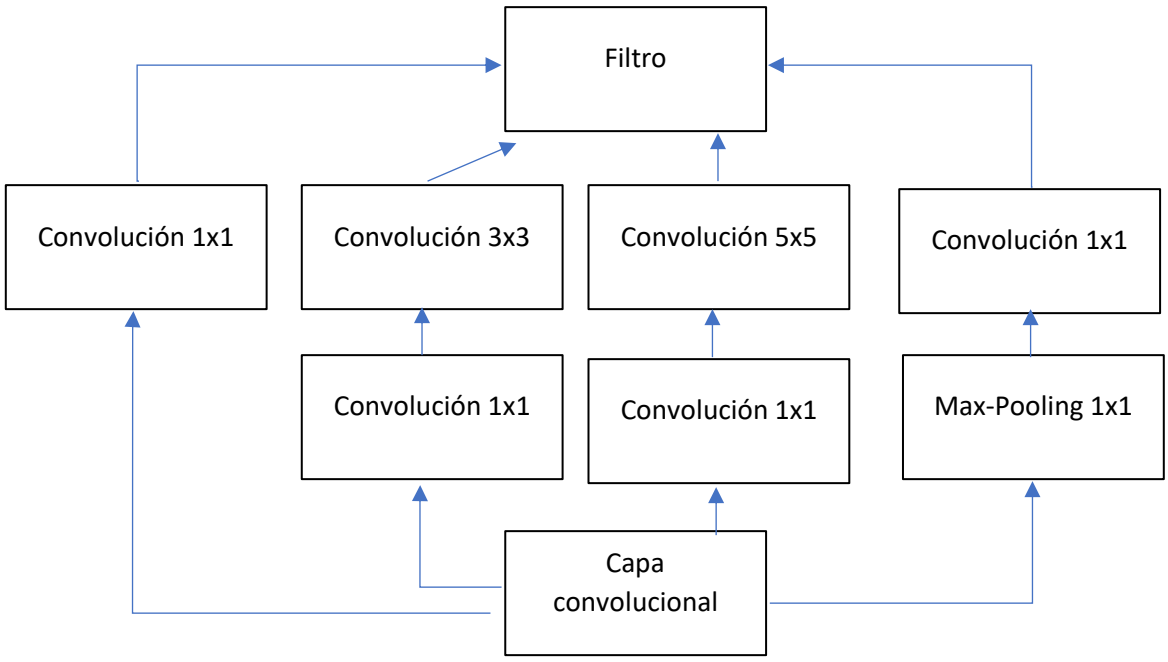


Fig. 3.23 Arquitectura de la red neuronal convolucional Inception_v3

A razón de obtener el mejor desempeño de las redes neuronales convolucionales se detallaron diferentes pruebas, ya que representan diferentes arquitecturas por lo que los resultados y el

desempeño cambia. A continuación, se describen las pruebas realizadas y los resultados obtenidos.

Prueba 1: En esta prueba se utilizaron 9 patrones con 200 imágenes cada uno con un total de 1800 instancias. En la Tabla 3.16, se registran los resultados obtenidos del entrenamiento con una configuración de 500 bach, un total de 100 épocas y un tiempo de entrenamiento de 6.3 horas, da una precisión de 88%. El objetivo de realizar esta prueba en la red neuronal convolucional de Inception_V3 es observar el comportamiento con respecto a AlexNet, los resultados obtenidos verifican que tiene la misma taza de error que la prueba 1 de AlexNet, los valores marcados en rojo son las precisiones correctas basados en encontrarse en los primeros cuatro lugares más cercanos a 100%.

Tabla 3.33 Resultados de Entrenamiento para Inception_V3 con 9 patrones y 200 instancias c/u

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker	Ranking
Chat	1	0.16358	0.04076	0.10806	0.06382	0.01578	0.15495	0.14627	0.08844
Comment	0.03356	1	0.03502	0.01721	0.03612	0.04643	0.13155	0.12357	0.07479
Gallery	0.03578	0.11136	1	0.07059	0.03841	0.02072	0.02647	0.11297	0.22975
Menu-horizontal	0.05365	0.0202	0.0357	1	0.0262	0.00527	0.04673	0.04739	0.50558
Menu-vertical	0.02378	0.07978	0.12727	0.09396	0	0.02127	0.13903	0.11171	0.30199
Music player	0.00811	0.62126	0.04349	0.00424	0.0149	1	0.09181	0.07292	0.03582
Panel	0.07911	0.32926	0.05673	0.08863	0.1117	0.02002	1	0.11249	0.08259
Picker	0.0317	0.3057	0.09098	0.04878	0.02888	0.01698	0.0281	1	0.04972
Ranking	0.05903	0.12544	0.0694	0.0685	0.06121	0.01539	0.05159	0.15177	1

Prueba 2: En esta prueba se utilizaron 9 patrones con 400 imágenes cada uno con un total de 3800 imágenes. En la Tabla 3.17 se registran los resultados obtenidos del entrenamiento con una configuración de 500 bach, un total de 100 épocas y un tiempo de entrenamiento de 8.7 horas, da una precisión de 100%. Como objetivo principal de esta prueba es comparar los resultados con respecto a la anterior prueba de este modelo y del modelo con AlexNet. Es importante resaltar que en esta prueba se obtuvo una precisión del 100% con respecto a AlexNet, otro punto a resaltar es que se usó el proceso de transformación de la imagen.

Tabla 3.34 Tabla de entrenamiento para Inception_V3 con 9 patrones y 400 instancias c/u

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker	Ranking
Chat	1.0000	0.1099	0.0154	0.0741	0.1613	0.0248	0.3284	0.0428	0.1255
Comment	0.0110	1.0000	0.0202	0.0168	0.1982	0.0830	0.1502	0.0620	0.0327
Gallery	0.0209	0.0703	1.0000	0.0183	0.0493	0.0156	0.0414	0.0239	0.6514
Menu-horizontal	0.0379	0.0077	0.0124	1.0000	0.0734	0.0155	0.0566	0.0130	0.4272
Menu-vertical	0.0128	0.0435	0.0368	0.0382	1.0000	0.0198	0.3943	0.0196	0.2611
Music player	0.0037	0.5016	0.0189	0.0021	0.0456	1.0000	0.1216	0.0299	0.0260
Panel	0.0222	0.0710	0.0090	0.0266	0.1887	0.0149	1.0000	0.0115	0.0317
Picker	0.0067	0.3957	0.0251	0.0357	0.1394	0.0459	0.0441	1.0000	0.0219
Ranking	0.0463	0.0519	0.0267	0.0168	0.0712	0.0144	0.1262	0.0308	1.0000

Prueba 3: En esta prueba se utilizaron 17 patrones con 400 imágenes cada uno, cuenta con un total de 6800 instancias en total. En la Tabla 3.18 se registran los resultados obtenidos del entrenamiento con una configuración de 500 bach, un total de 100 épocas y un tiempo de entrenamiento de 17.3 horas, da una precisión de 83%. Con respecto a AlexNet y se observa que a pesar de que se aumentaron los UIDPs la precisión no fue tan negativa con respecto a AlexNet, esto es por la disminución de errores que existe con la red neuronal de Inception_v3, por lo tanto se menciona que la prueba resalta una buena precisión con respecto a AlexNet.

Tabla 3.35 Tabla de entrenamiento para Inception_V3 con 17 patrones y 400 instancias c/u (1)

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker
Chat	0.00000	0.08880	0.01870	0.05369	0.10847	0.02138	0.18422	0.03406
Comment	0.01160	1.00000	0.02795	0.01246	0.10337	0.07369	0.11080	0.05399
Gallery	0.02634	0.07821	1.00000	0.02130	0.04466	0.02238	0.04868	0.02750
Menu-horizontal	0.04045	0.01183	0.01511	1.00000	0.06968	0.01361	0.05984	0.01513
Menu-vertical	0.01634	0.04984	0.03953	0.03876	1.00000	0.02447	0.29296	0.02329
Music player	0.00430	0.46879	0.03094	0.00195	0.03220	1.00000	0.10284	0.03282
Panel	0.01786	0.06664	0.01285	0.02054	0.11705	0.01557	1.00000	0.01199
Picker	0.00648	0.25056	0.03011	0.01811	0.07232	0.04055	0.03278	1.00000

Tab.3.18 Tabla de entrenamiento para Inception_V3 con 17 patrones y 400 instancias c/u (2)

	Ranking	smcomposicion1	smcomposicion2	smcomposicion3	smcomposicion4	smcomposicion5	smcomposicion6	smcomposicion7	smcomposicion8
Ranking	1.00000	0.03395	0.01339	0.00869	0.00784	0.01237	0.03815	0.01903	0.01710
smcomposicion1	0.21545	1.00000	0.04459	0.02661	0.01480	0.01628	0.04332	0.02547	0.02157
smcomposicion2	0.07749	0.09150	0.00000	0.00996	0.01944	0.05044	0.08482	0.08136	0.04509
smcomposicion3	0.02678	0.07313	0.07401	1.00000	0.02536	0.03803	0.04841	0.06709	0.02168
smcomposicion4	0.03589	0.03395	0.01339	0.00869	1.00000	0.01237	0.03815	0.01903	0.01710
smcomposicion5	0.21545	0.01911	0.04459	0.02661	0.01480	1.00000	0.04332	0.02547	0.02157
smcomposicion6	0.07749	0.09150	0.02168	0.00996	0.01944	0.05044	0.00000	0.08136	0.04509
smcomposicion7	0.49122	0.02814	0.01520	0.01650	0.00578	0.00793	0.02172	1.00000	0.00848
smcomposicion8	0.34856	0.02146	0.02168	0.04673	0.01455	0.00659	0.00672	0.00755	1.00000

Prueba 4: En esta prueba se utilizaron 27 patrones con 400 imágenes cada uno por lo tanto da un total de 10,800 imágenes. En la Tabla 3.19 se registran los resultados obtenidos del entrenamiento con una configuración de 500 bach, un total de 100 épocas y con un tiempo de entrenamiento de 34.3 horas, da una precisión de 74%. Para esta prueba 4 se observan varios detalles, una excelente mejora con respecto a la misma cantidad entrenada en el modelo AlexNet, ya que a pesar de que se cuentan con la misma cantidad de imágenes para cada UIDP la precisión se sigue manteniendo con respecto a la tercera prueba generada en Inception_v3. Por lo tanto, esta red funciona mucho mejor con respecto a AlexNet.

Tabla 3.36 Tabla de entrenamiento de Inception_v3 con 27 patrones (1)

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker	Ranking	smcomposicion1	smcomposicion2	smcomposicion3	smcomposicion4	smcomposicion5
Chat	0.00000	0.08880	0.01870	0.05369	0.10847	0.02138	0.18422	0.03406	0.10925	0.03590	0.02242	0.01740	0.01664	0.02940
Comment	0.01160	1.00000	0.02795	0.01246	0.10337	0.07369	0.11080	0.05399	0.03768	0.03761	0.05748	0.01738	0.02065	0.03676
Gallery	0.02634	0.07821	1.00000	0.02130	0.04466	0.02238	0.04868	0.02750	0.49122	0.02814	0.01520	0.01650	0.00578	0.00793
Menu-horizontal	0.04045	0.01183	0.01511	1.00000	0.06968	0.01361	0.05984	0.01513	0.34856	0.02146	0.02168	0.04673	0.01455	0.00659
Menu-vertical	0.01634	0.04984	0.03953	0.03876	1.00000	0.02447	0.29296	0.02329	0.21972	0.04172	0.02214	0.01202	0.00559	0.00696

Tabla 3.37 Tabla de entrenamiento de Inception_v3 con 27 patrones (1)

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker	Ranking	smcomposicion1	smcomposicion2	smcomposicion3	smcomposicion4	smcomposicion5
Music player	0.00430	0.46879	0.03094	0.00195	0.03220	1.00000	0.10284	0.03282	0.03589	0.01911	0.01268	0.01018	0.00370	0.00528
Panel	0.01786	0.06664	0.01285	0.02054	0.11705	0.01557	1.00000	0.01199	0.03860	0.08717	0.02580	0.00922	0.01193	0.01810
Picker	0.00648	0.25056	0.03011	0.01811	0.07232	0.04055	0.03278	1.00000	0.02678	0.07313	0.07401	0.01405	0.02536	0.03803
Ranking	0.04418	0.06247	0.03536	0.01912	0.06569	0.01837	0.11153	0.03441	1.00000	0.03395	0.01339	0.00869	0.00784	0.01237
smcomposicion1	0.01727	0.09731	0.04375	0.04408	0.11661	0.04019	0.07989	0.02728	0.21545	1.00000	0.04459	0.02661	0.01480	0.01628
smcomposicion2	0.01415	0.01971	0.00689	0.04201	0.31074	0.00408	0.07559	0.01574	0.07749	0.09150	0.00000	0.00996	0.01944	0.05044
smcomposicion3	0.00648	0.25056	0.03011	0.01811	0.07232	0.04055	0.03278	0.03282	0.02678	0.07313	0.07401	1.00000	0.02536	0.03803

Tabla 3.38 Tabla de entrenamiento de Inception_v3 con 27 patrones (1)

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker	Ranking	smcomposicion1	smcomposicion2	smcomposicion3	smcomposicion4	smcomposicion5
smcomposicion4	0.04418	0.06247	0.03536	0.01912	0.06569	0.01837	0.11153	0.03441	0.03589	0.03395	0.01339	0.00869	1.00000	0.01237
smcomposicion5	0.01727	0.09731	0.04375	0.04408	0.11661	0.04019	0.07989	0.02728	0.21545	0.01911	0.04459	0.02661	0.01480	1.00000
smcomposicion6	0.01415	0.01971	0.00689	0.04201	0.31074	0.00408	0.07559	0.01574	0.07749	0.09150	0.02168	0.00996	0.01944	0.05044
smcomposicion7	0.02634	0.07821	0.01971	0.02130	0.04466	0.02238	0.04868	0.02750	0.49122	0.02814	0.01520	0.01650	0.00578	0.00793
smcomposicion8	0.04045	0.01183	0.01511	0.05984	0.06968	0.01361	0.05984	0.01513	0.34856	0.02146	0.02168	0.04673	0.01455	0.00659
mcomposicion1	0.03212	0.01154	0.11234	0.07680	0.04670	0.04798	0.07619	0.04758	0.07686	0.27643	0.08840	0.05869	0.37675	0.03975

Tabla 3.39 Tabla de entrenamiento de Inception_v3 con 27 patrones (1)

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker	Ranking	smcomposicion1	smcomposicion2	smcomposicion3	smcomposicion4	smcomposicion5
mcomposicion2	0.06322	0.09462	0.01658	0.00645	0.06165	0.14647	0.01277	0.01640	0.07560	0.01647	0.04166	0.04657	0.33254	0.05627
mcomposicion3	0.01549	0.12457	0.09158	0.02430	0.02468	0.04968	0.04616	0.01649	0.04573	0.03266	0.06580	0.11341	0.00994	0.05846
mcomposicion4	0.00564	0.09915	0.06514	0.03417	0.07816	0.06779	0.01654	0.01976	0.07678	0.06434	0.05400	0.08601	0.05791	0.02180
mcomposicion5	0.01225	0.04665	0.00646	0.04457	0.09679	0.14673	0.01776	0.01550	0.03164	0.04357	0.04786	0.02712	0.07024	0.08668
mcomposicion6	0.04887	0.08785	0.08910	0.01677	0.07461	0.04647	0.07914	0.05476	0.01647	0.04677	0.05639	0.01335	0.05651	0.08636
mcomposicion7	0.00781	0.11548	0.01345	0.01247	0.07691	0.04677	0.06477	0.04973	0.01679	0.04677	0.03833	0.00751	0.08764	0.08894

Tabla 3.40 Tabla de entrenamiento de Inception_v3 con 27 patrones (1)

	Chat	Comment	Gallery	Menu-horizontal	Menu-vertical	Music player	Panel	Picker	Ranking	smcomposicion1	smcomposicion2	smcomposicion3	smcomposicion4	smcomposicion5
mcomposicion8	0.00046	0.04879	0.04676	0.07799	0.04645	0.01675	0.01967	0.09183	0.01576	0.06756	0.00107	0.06474	0.04630	0.06451
mcomposicion9	0.00955	0.08089	0.01654	0.05167	0.03497	0.04796	0.01673	0.01640	0.04642	0.05397	0.03068	0.07057	0.06276	0.03966
mcomposicion10	0.02335	0.00780	0.03255	0.03467	0.01546	0.04675	0.01460	0.01676	0.01646	0.03202	0.00305	0.01691	0.04940	0.06432

Tabla 3.19 Tabla de entrenamiento de Inception_v3 con 27 patrones (2)

	smcomposicion6	smcomposicion7	smcomposicion8	mcomposicion1	mcomposicion2	mcomposicion3	mcomposicion4	mcomposicion5	mcomposicion6	mcomposicion7	mcomposicion8	mcomposicion9	mcomposicion10
Chat	0.05851	0.06019	0.05348	0.08765	0.00121	0.02687	0.05679	0.07132	0.05964	0.02585	0.04931	0.03952	0.07604
Comment	0.02832	0.05689	0.01749	0.00578	0.09306	0.00284	0.01789	0.05628	0.06195	0.04125	0.00789	0.02132	0.09536
Gallery	0.02172	0.01205	0.00848	0.04603	0.06734	0.00997	0.04488	0.02036	0.03665	0.06875	0.07734	0.03977	0.09203

Tabla 3.19 Tabla de entrenamiento de Inception_v3 con 27 patrones (2)

	smcomposicion6	smcomposicion7	smcomposicion8	mcomposicion1	mcomposicion2	mcomposicion3	mcomposicion4	mcomposicion5	mcomposicion6	mcomposicion7	mcomposicion8	mcomposicion9	mcomposicion10
Menu-horizontal	0.00672	0.00755	0.06205	0.08850	0.08647	0.02771	0.01700	0.00246	0.07003	0.03067	0.07468	0.03132	0.00276
Menu-vertical	0.03450	0.02044	0.01174	0.06300	0.06484	0.03947	0.07306	0.08343	0.00341	0.00946	0.07732	0.02321	0.07589
Music player	0.01160	0.01060	0.00725	0.09437	0.04262	0.00699	0.00389	0.01673	0.01562	0.00632	0.02263	0.02636	0.04718
Panel	0.14658	0.07344	0.03050	0.09801	0.01061	0.05644	0.02622	0.06368	0.02769	0.09737	0.03040	0.05082	0.03886
Picker	0.04841	0.06709	0.02168	0.02384	0.01440	0.00661	0.01924	0.05995	0.09904	0.05146	0.06477	0.09414	0.04609
Ranking	0.03815	0.01903	0.01710	0.07344	0.08870	0.00658	0.09608	0.06571	0.01462	0.02713	0.06913	0.07768	0.00679
smcomposicion1	0.04332	0.02547	0.02157	0.02496	0.01632	0.07483	0.01298	0.05065	0.06749	0.05420	0.06388	0.00494	0.03040
smcomposicion2	0.08482	0.08136	0.04509	0.08765	0.05854	0.06787	0.06222	0.00601	0.09556	0.06052	0.02445	0.02835	0.01376

Tabla 3.19 Tabla de entrenamiento de Inception_v3 con 27 patrones (2)

	smcomposicion6	smcomposicion7	smcomposicion8	mcomposicion1	mcomposicion2	mcomposicion3	mcomposicion4	mcomposicion5	mcomposicion6	mcomposicion7	mcomposicion8	mcomposicion9	mcomposicion10
smcomposicion3	0.04841	0.06709	0.02168	0.04654	0.08238	0.03799	0.02631	0.07414	0.04873	0.08429	0.04182	0.06889	0.01540
smcomposicion4	0.03815	0.01903	0.01710	0.03150	0.01726	0.04940	0.03364	0.08384	0.05250	0.01689	0.07871	0.03022	0.07019
smcomposicion5	0.04332	0.02547	0.02157	0.04333	0.04651	0.03096	0.00809	0.08511	0.04500	0.07765	0.00255	0.08014	0.05713
smcomposicion6	0.00000	0.08136	0.04509	0.09582	0.06079	0.07344	0.05576	0.01322	0.03437	0.07599	0.14040	0.03170	0.01407
smcomposicion7	0.02172	1.00000	0.00848	0.04539	0.07144	0.01200	0.03013	0.08158	0.07015	0.02314	0.05791	0.06923	0.07725
smcomposicion8	0.00672	0.00755	0.00000	0.08399	0.04891	0.06016	0.01509	0.09061	0.08236	0.05839	0.07888	0.08243	0.00448

Tabla 3.19 Tabla de entrenamiento de Inception_v3 con 27 patrones (2)

	smcomposicion6	smcomposicion7	smcomposicion8	mcomposicion1	mcomposicion2	mcomposicion3	mcomposicion4	mcomposicion5	mcomposicion6	mcomposicion7	mcomposicion8	mcomposicion9	mcomposicion10
mcomposicion1	0.01743	0.05022	0.08788	1.00000	0.02751	0.00284	0.01470	0.03925	0.07445	0.07210	0.03864	0.04426	0.09594
mcomposicion2	0.00486	0.05893	0.05168	0.07646	1.00000	0.09141	0.08356	0.03557	0.01158	0.03131	0.09313	0.06678	0.09981
mcomposicion3	0.02661	0.05078	0.03947	0.05961	0.05708	1.00000	0.09310	0.09077	0.05868	0.03530	0.09766	0.02952	0.02635
mcomposicion4	0.43463	0.03641	0.11439	0.09236	0.02278	0.04775	1.00000	0.01808	0.00924	0.00537	0.04612	0.04691	0.09201
mcomposicion5	0.06957	0.05221	0.00491	0.08219	0.06810	0.02352	0.04580	1.00000	0.08084	0.07090	0.09129	0.26731	0.05046
mcomposicion6	0.09025	0.06138	0.06388	0.09385	0.03192	0.00799	0.04422	0.09007	0.00000	0.06350	0.01096	0.02519	0.08074

Tabla 3.19 Tabla de entrenamiento de Inception_v3 con 27 patrones (2)

	smcomposicion6	smcomposicion7	smcomposicion8	mcomposicion1	mcomposicion2	mcomposicion3	mcomposicion4	mcomposicion5	mcomposicion6	mcomposicion7	mcomposicion8	mcomposicion9	mcomposicion10
mcomposicion7	0.04777	0.06501	0.04288	0.04502	0.03810	0.09884	0.00898	0.06669	0.08241	1.00000	0.08560	0.07396	0.08150
mcomposicion8	0.01230	0.02552	0.06528	0.07417	0.09469	0.06402	0.08594	0.06195	0.09583	0.09383	0.00000	0.03787	0.07914
mcomposicion9	0.02272	0.21662	0.00788	0.06311	0.00554	0.06046	0.00964	0.00265	0.08089	0.14421	0.00891	1.00000	0.00747
mcomposicion10	0.06531	0.02157	0.03991	0.02432	0.08786	0.09430	0.02943	0.00638	0.01658	0.01419	0.08947	0.06854	0.00000

En conclusión, se observa que en las diferentes pruebas los datos que cambian son: la cantidad de patrones y el número de instancias. Respecto a las primeras pruebas los resultados obtenidos determinaron el correcto funcionamiento de la red neuronal convolucional y se garantizó una adecuada clasificación, con lo que se obtuvo un margen de referencia que permitió ampliar la investigación y abrir la oportunidad para crear un nuevo estándar o enfoque para el desarrollo de software. Lo anterior, considera que el tiempo promedio de respuesta es menor a 60 segundos. Con tal resultado se observó claramente el impacto que genera una red neuronal convolucional para la clasificación de imágenes.

Respecto a los datos obtenidos con la red convolucional Inception_V3 se observa una mejoría en la clasificación de imágenes con un mínimo de 30.7% más en comparación con AlexNet. Además, Inception_v3 tiene una mejoría clara con respecto a su margen de error de 6.67% con respecto a AlexNet que cuenta con un 15.3%, la respuesta notable de Inception_V3 es de 18% en comparación con AlexNet. Cabe señalar que la implementación de la API Keras facilita la programación al contar con módulos de funciones predefinidos. Otro punto a considerar es el uso computacional, en Inception_v3 es menor que en AlexNet, se utiliza menos recursos de memoria y genera un porcentaje de respuesta más rápido que AlexNet conforme a la clasificación. Por otra parte, Inception_v3 cuenta con un mayor soporte a la comunidad por ser un modelo más reciente y usado por los científicos.

3.5.Implementación del módulo del procesamiento

En este punto se describen detalles importantes de la red neuronal convolucional, se especifican los diagramas de clase y el código referente a sus funciones principales.

3.5.1. Diagramas de clases

Para los diagramas de clases se dividen en dos partes: entrenamiento y su exportación, y diagrama de ejecución del servicio REST. En la Fig. 7 se presenta el modelo de Inpcetion_V3 que parte del entrenamiento, donde se observan las diferentes clases tales como: `EntrenamientoCNN`, encargada principalmente de cargar componentes de la red neuronal y atributos que se necesitan para manipular y generar un modelo adecuado de entrenamiento. Keras, numpy, tensorflow y pyplot son clases que cargan diferentes métodos. En el caso de Keras cuenta con los métodos de: `inceptionv3 (), activation (), BGD (), Dropout (), Flatter () y Dense (),` métodos esenciales para el entrenamiento de la red. La clase Transformación importa las clases de cv2, os y sys, además cuenta con los métodos de `rgb2gray ()` que solicitan una variable de tipo rgb y `leet ()`. La clase `ExportarModelo` encargada principalmente en generar un modelo basado en keras, un archivo del modelo generado el cual se aprovecha para cargar en un servidor, cuenta con las clases de tensorflow y numpy para su implementación, cuenta con atributos especiales de

carga del nombre del modelo, la firma generada del modelo, dirección de la exportación y el constructor del modelo (ver Fig. 3.24).

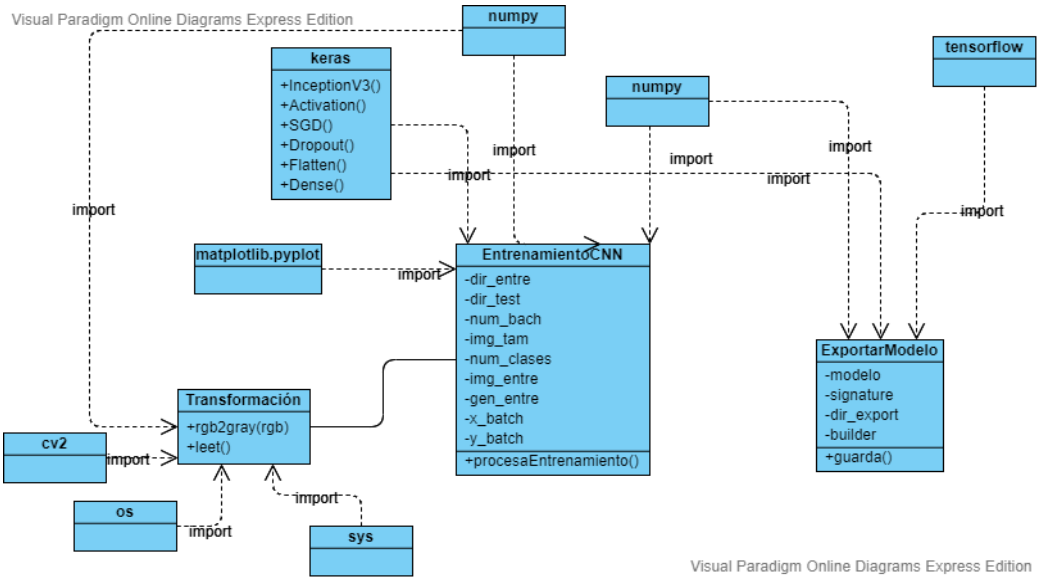


Fig. 3.24 Diagrama de clases para el entrenamiento

Para el diagrama de clase de servidor encargado específicamente para ejecutar el servicio y transformar la instancia cada vez que se solicita la implementación del servicio.

En la clase `Run_server` contiene el atributo app y el método `image_classifier`, además se importan diferentes clases de keras, flask, base64, json, io, numpy y request. Se cargan los métodos especiales de Inpcetionv3, `image, flask y jsonify y BytesIO` (ver Fig. 3.25).

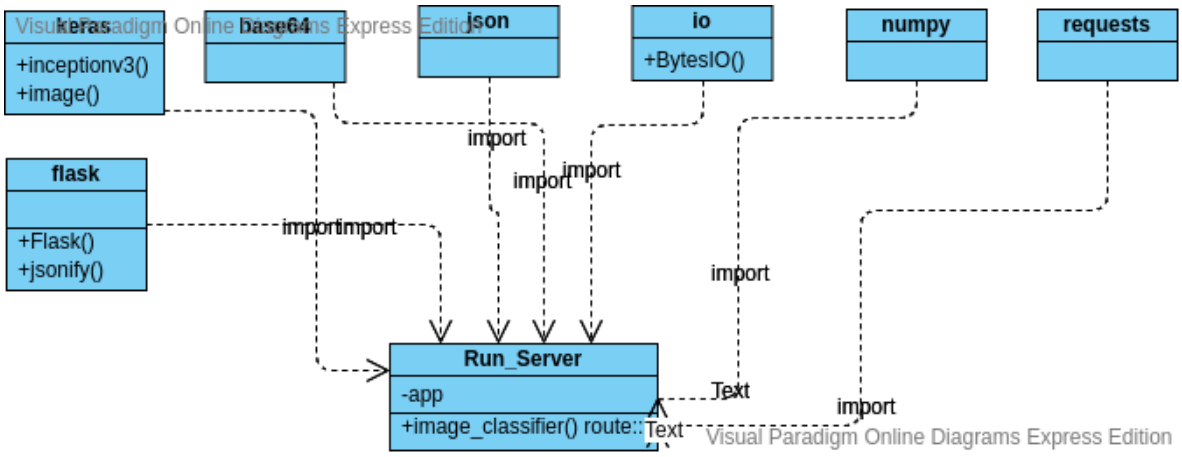


Fig. 3.25 Diagrama de clases para cargar el servicio REST

3.5.2. Funcionalidades principales

En este punto se describen clases importantes del funcionamiento de la red neuronal convolucional, compuesto por la clase `Entrenamiento.py`. En este sentido, el método *`rgb2gray()`* se encarga principalmente de obtener el arreglo de la imagen, el método *`leet()`* es el encargado de leer todas las carpetas del conjunto de los datos, una vez que lee cada carpeta inicia la creación de una nueva carpeta que contiene un nuevo conjunto de imágenes, genera una nueva matriz conforme a los algoritmos de *dilatación*, *medianBlur*, *normalización* y *otsu*, guarda los archivos en la nueva carpeta e imprime los resultados de cada transformación de la imagen (ver Listado de código. 3.1)

Listado de código. 3.1 Clase Transformacion.py

```
1. *- coding: utf-8 -*-
2. 
3. """
4. Created on Wed Aug 28 11:47:42 2019
5. @author: aaron
6. """
7. 
8. import os
9. import sys
10. import cv2
11. import numpy as np
12. 
```

```
13. class Transformacion :
14. def rgb2gray(rgb):
15.     return np.dot(rgb[...,:3], [0.2989, 0.5870, 0.1140])
16. 
17. def leet():
18.     try:
19.         dir_datos  = str(sys.argv[1].rstrip('/'))  #Dirección
20.         tam_img   = str(sys.argv[2])
21.         dir_datos_finales  = str(sys.argv[3].rstrip('/')) #Salida del directorio
22.         print ("Inicia")
23.         print ("Conjunto de datos en  %s " % dir_datos)
24.         tclass = [ d for d in os.listdir( dir_datos ) ]
25.         contador = 0
26.         for x in tclass:
27.           list_dir =  os.path.join(dir_datos, x )
28.           list_tuj = os.path.join(dir_datos_finales+'/', x+'/')
29. 
30.           if not os.path.exists(list_tuj):
31.               os.makedirs(list_tuj)
32.           if os.path.exists(list_tuj):
33.              for i in os.listdir(list_dir):
34.                 try:
35.                     print ("entra")
36.                     img = cv2.imread(os.path.join(input_dir+'/'+x,i), -1)
37.                     rgb_planes = cv2.split(img)
38.                     arr = np.array([])
39.                     result_norm_planes = []
40.                     for plane in rgb_planes:
41.                      #Transformación de la imagen
42.                         dilated_img = cv2.dilate(plane, np.ones((7,7), np.uint8))
43.                         bg_img = cv2.medianBlur(dilated_img, 21)
44.                         diff_img = 255 - cv2.absdiff(plane, bg_img)
45.                         norm_img = cv2.normalize(diff_img,arr, alpha=0, beta=255, norm
    _type=cv2.NORM_MINMAX, dtype=cv2.CV_8UC1)
46.                         result_norm_planes.append(norm_img)
47.                     result_norm = cv2.merge(result_norm_planes)
48.                     result_norm = rgb2gray(result_norm)  # Normaliza imagen
49.                     threshold_global_otsu = threshold_otsu(result_norm)
50.                     global_otsu = img >= threshold_global_otsu
51. 
52.                     fname,extension = os.path.splitext(i)
53.                     nuevo_archivo = fname+extension
54.                     if extension != ".jpg" :
55.                         newfile = fname + ".jpg"
56.                     cv2.imwrite(os.path.join(dir_datos_finales+'/'+x,nuevo_archivo), glo
    bal_otsu, [int(cv2.IMWRITE_JPEG_QUALITY), 90])
57.                     print ("Transformación de la imagen: %s - %s " % (x,i))
```

```
                except Exception:
                    print ("Error al transformar la imagen: %s - %s " % (x,i))
                    sys.exit(1)
            contador +=1
    except Exception:
        print ("Error, del directorio: ")
        sys.exit(1)
leet()

```

De la clase entrenamiento se describen las siguientes características:

- Utiliza la biblioteca para evaluar las imágenes.
- Se especifican directorios de las imágenes y se agregan atributos requeridos.
- El método procesaEntrenamiento es la encargada de validar imágenes y cargar el modelo Inception_V3 entrenado en imagenet para ejecutar el entrenamiento.
- Congela el modelo, es decir permite obtener sus pesos y atributos de la red.
- Crea modelo y se agregan bases que conformaran la red convolucional.
- Imprime detalles del modelo.
- Compila el modelo.
- Ejecuta el entrenamiento conforme a los atributos específicos como batch y épocas.
- Guarda el modelo.
- Imprime estadísticas del proceso de entrenamiento.

La representación de la clase ejecutada se muestra en el listado código 3.2.

Listado de código. 3.2 Clase Entrenamiento.py

```
# -*- coding: utf-8 -*-

"""
Created on Wed Aug 28 11:47:42 2019

@author: aaron
"""

from keras.preprocessing.image import ImageDataGenerator
```

```
import numpy as np
import matplotlib.pyplot as plt
import keras
from keras.applications import InceptionV3
from keras.layers import Activation, Dropout, Flatten, Dense
from keras.optimizers import SGD

class Entrenamiento:

dir_entre = 'dataset/'
dir_test = 'test/'
num_bach = 500
img_tam = 255
num_clases = 27

img_entre = ImageDataGenerator(validation_split=0.3,
                               shear_range=0.2,
                               zoom_range=0.2,
                               horizontal_flip=True)

gen_entre = img_entre.flow_from_directory(
                    directory=dir_entre,
                    target_size=(img_tam,img_tam),
                    batch_size=num_bach,
                    class_mode='categorical',
                    color_mode='rgb',
                    shuffle=True)

x_batch, y_batch = gen_entre.next()

def procesaEntrenamiento()
# Imprime las imagenes, verifica la validación de las imagenes
fig=plt.figure()
columna = 5
fila = 5
for i in range(1, columna*fila):
    num = np.random.randint(num_bach)
    imagen = x_batch[num].astype(np.int)
    fig.add_subplot(fila, columna, i)
    plt.imshow(imagen)
plt.show()

#Carga el modelo de Inception_v3
modelo_base = InceptionV3(weights='imagenet', include_top=False, input_shape=(i
mg_tam, img_tam, 3))

print(modelo_base.summary())
```

```
56.
57. # Congela el modelo
58. for capa in modelo_base.layers:
59.     capa.trainable = False
60.
61. # Crea el modelo
62. modelo = keras.models.Sequential()
63.
64. # Agrega el modelo de convolución
65. modelo.add(modelo_base)
66.
67. # Agrega nueva base
68. modelo.add(Flatten())
69. modelo.add(Dense(1024, activation='relu'))
70. modelo.add(Dense(num_clases, activation='softmax'))
71.
72. # Imprime los detalles del modelo
73. print(modelo.summary())
74.
75. #  Compile the model
76. modelo.compile(loss='categorical_crossentropy',
77.           optimizer=SGD(lr=1e-3),
78.           metrics=['accuracy'])
79.
80. #  Inicia el proceso de entrenamiento
81. historico = modelo.fit_generator(
82.         gen_entre,
83.         steps_per_epoch=gen_entre.n/num_bach,
84.         epochs=100)
85.
86. # Guarda el modelo con el siguiente nombre
87. modelo.save('modelo_final.h5')
88.
89. # Imprime el resumen historico del modelo
90. plt.plot(historico.history['loss'])
91. plt.title('loss')
92. plt.ylabel('loss')
93. plt.xlabel('epoch')
94. plt.legend(['loss'], loc='upper left')
95. plt.show()
```

En la clase Exportar Modelo representado en Listado de código.3.3 permite ordenar los modelos en diferentes carpetas necesario para el versionado de software, cuando se generen

nuevos reentrenamientos para aumentar las imágenes y tener una recopilación de los modelos adquiridos y entrenados.

Cuenta con la siguiente estructura:

- Importación de atributos y métodos.
- Carga el nombre del modelo
- Firmas de predicción, encargadas de medir los pesos y variables del modelo
- Genera un modelo de prueba y el modelo conforme a la dirección que se solicita.
- Guarda el modelo en archivos específicos para su correcta implementación.

Listado de código. 3.3 Clase ExportarModelo.py

```
# -*- coding: utf-8 -*-
"""
Created on Wed Aug 28 11:47:42 2019

@author: aaron
"""

from keras.models import load_model
import numpy as np
from keras import backend as K
import tensorflow as tf

class ExportarModelo :
modelo = load_model('modelo_final.h5')
signature = tf.saved_model.signature_def_utils.predict_signature_def(
    inputs={'image': modelo.input}, outputs={'scores': modelo.output})

dir_export = 'my_image_classifier/1'

builder = tf.saved_model.builder.SavedModelBuilder('tmp/my_saved')

def guarda() :

builder.add_meta_graph_and_variables(
    sess=K.get_session(),
    dir_export,
    tags=[tf.saved_model.tag_constants.SERVING],
    signature_def_map={
```

```
29.         tf.saved_model.signature_constants.DEFAULT_SERVING_SIGNATURE_D
    EF_KEY:
30.             signature
31.     })
32. builder.save()
33. print ('Guardado satisfactoriamente')
```

Para la clase servidor es necesario comprender ciertas definiciones que se le atribuyen a la construcción del servicio. Como primer punto se definen ciertos componentes que se necesitan para entender la arquitectura. El servicio REST (*Representational State Transfer*, Transferencia de Estado Representacional), permite crear un estilo de arquitectura con ciertas restricciones y respetar claramente el protocolo HTTP. *TensorFlow Serving* es un sistema de servicio flexible y de alto rendimiento para modelos de aprendizaje automático, diseñado especialmente para entornos productivos. Además, *TensorFlow Serving* hace mucho más sencillo la implementación de nuevos algoritmos y experimentos, manteniendo la misma API y arquitectura, provee una alta integración con los modelos generados por la API *TensorFlow*, inclusive con otros tipos de modelos. *Servables* es la extracción central en *TensorFlow Serving*, se encargan de hacer cálculos entre actividades que marcan: resultados de transmisión, experimentos de la API, operación asíncrona y no mantienen su propio ciclo de vida. *Servable Versions* permite cargar más configuraciones de algoritmos. *Streams Servibles*, ordena las versiones, *Models* representan uno o más servidores, *Loaders*, gestionan el API para cargar y descargar un servidor. Fuente, permite cero o más secuencias de servicios de los *Loader*, *Managers* manejan el clico de vida de los servidores; y *Core*, gestiona los *Servables* (ver. Fig. 3.26).

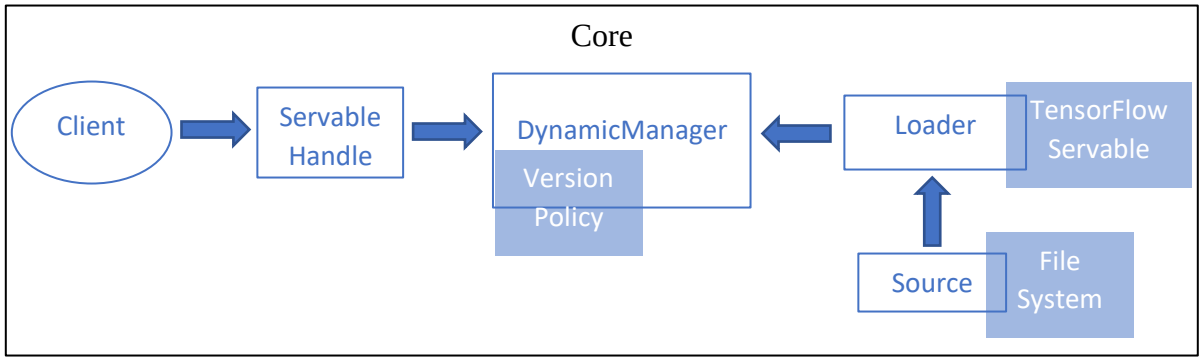


Fig. 3.26 Arquitectura de TensorFlow Serving

Sources crea *Loaders* para los *Servable Versions*, mientras que *Loaders* enviados como *Aspire Version*, quienes atienden las solicitudes de los clientes. Se toma en cuenta que la imagen pasa en una cadena codificada en Base64. Ya que JSON no tiene otra forma de especificar la imagen ya que es una matriz. La tecnología se aplica en Python 3.6, anaconda 1.9.6 y TensorFlow al iniciar se aplicará sobre el puerto 8501 y la API REST se localizará en el puerto 9000. Al solicitar la respuesta se verifica que se encuentra totalmente estable y funcional de la siguiente forma (ver Fig. 3.27). Por último, se necesita ejecutar un comando específico en la terminal para iniciar el servicio que cuenta con la siguiente codificación (ver Listado de código.3.4).

Listado de código. 3.4 Comando para la terminal en linux

1. -$tensorflow_model_server --model_base_path=Dirección de la clase de Python
2. –rest_api_port=Puerto --model_name=Nombre del modelo

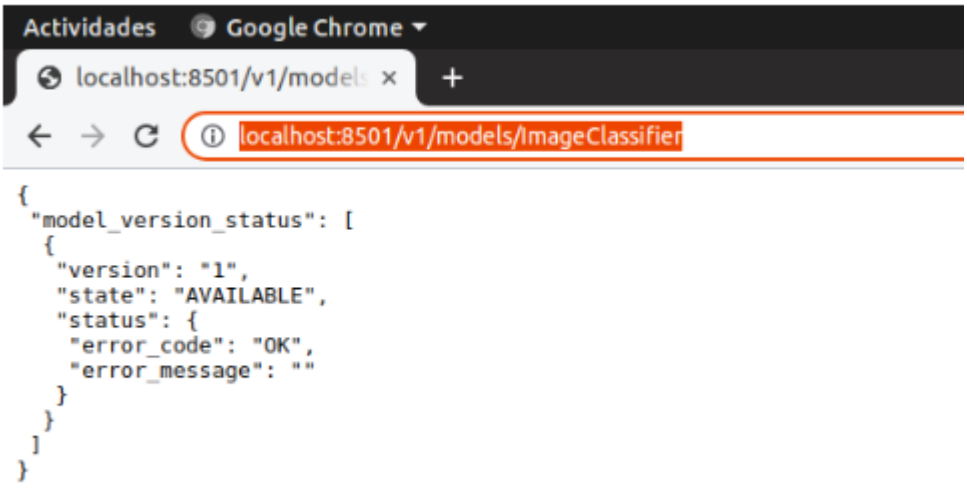


Fig. 3.27 Respuesta del servicio

Una vez generado esta opción permite mantener activo el servicio y obtener la clasificación necesaria conforme a la imagen solicitada (ver Listado de código.3.5).

La clase cuenta con la siguiente implementación:

- Ruta del clasificador
- Carga de la imagen,
- Transformación de la imagen.
- Generación de la estructura JSON
- Crea predicción en la ruta con formato utf-8
- Regresa las cuatro primeras posiciones clasificadas en JSON

Listado de código. 3.5 Clase Servidor.py

```
1. # -*- coding: utf-8 -*-
2. """
3. Created on Wed Aug 28 11:47:42 2019
4.
5. @author: aaron
6. """
7. import base64
8. import json
9. from io import BytesIO
10.
11. import numpy as np
12. import requests
13. from flask import Flask, request, jsonify
14. from keras.applications import inception_v3
15. from keras.preprocessing import image
16.
17. # Agrega la biblioteca Flask
18.
19. app = Flask(__name__)
20.
21. # se hace post sobre el servicio
22.
23. @app.route('/imageclassifier/predict/', methods=['POST'])
24. def image_classifier():
25.
26.     # Decodificaciòn y revisión de la imagen en base 64
27.     img = image.img_to_array(image.load_img(BytesIO(base64.b64decode(request.f
    orm['b64'])),
28.                                             target_size=(224, 224))) / 255.
29.     #tranformación por posible error y ejecución de transformación
30.     img = img.astype('float16')
31.     dilated_img = cv2.dilate(img, np.ones((7,7), np.uint8))
32.      bg_img = cv2.medianBlur(dilated_img, 21)
```

```
33.     diff_img = 255 - cv2.absdiff(plane, bg_img)
34.     norm_img = cv2.normalize(diff_img,arr, alpha=0, beta=255,       norm_type=cv2
    .NORM_MINMAX, dtype=cv2.CV_8UC1)
35.     result_norm_planes.append(norm_img)
36.     result_norm = cv2.merge(result_norm_planes)
37.     result_norm = rgb2gray(result_norm)  # Normaliza imagen
38.     threshold_global_otsu = threshold_otsu(result_norm)
39.      global_otsu = img >= threshold_global_otsu
40.     img = global_ otsu
41.
42.     # Crea la estructura de Json
43.     JsonCarga = {
44.         "instances": [{'input_image': img.tolist()}]
45.     }
46.
47.     # Making POST request
48.     r = requests.post('http://localhost:9000/v1/models/ImageClassifier:predict', json=J
    sonCarga)
49.
50.     # Decodificación de los resultados del servicio
51.     pred = json.loads(r.content.decode('utf-8'))
52.
53.     # Regresa la respuesta en JSON al FrontEnd los cuatro primeros
54.     return jsonify(inception_v3.decode_predictions(pred, top=4)))
```

En conclusión, conocer los diagramas de clases y el código fuente permite identificar partes importantes del desarrollo, como la implementación del entrenamiento y la ejecución de su servicio.

3.6. Implementación del módulo de generación de código.

En este capítulo se describe la constitución del módulo generador de código.

El generador se encuentra diseñado con el lenguaje de programación PHP, esto permite ser desplegado en cualquier tipo de servidor Web, como también la utilización de una posible respuesta RESTFul.

3.6.1. Archivo de configuración basado en XML

En este segmento se aborda la conformación de la estructura del archivo XML (ver Fig.3.28), de acuerdo a al formato y sus características, los datos principales de la aplicación son:

- *userName*: constituido por el nombre del usuario en la aplicación.
- *name*: atributo que describe el nombre de la aplicación.
- *vendor*: atributo del nombre de la institución o empresa.
- *shortName*: atributo del nombre corto de la aplicación.
- *version*: atributo que especifica la versión del proyecto de la aplicación.
- *icon*: atributo que corresponde al nombre de la imagen de la aplicación.
- *typeApp*: atributo donde se especifica el tipo de aplicación a generar en este caso una aplicación *e-Commerce* o *Social Media*.
- *authors*: etiqueta que permite almacenar los datos del autor de la aplicación como nombre, correo, sitio Web y por último el nombre de la compañía.
- *deploySettings*: un conjunto de atributos que permiten almacenar las características de despliegue de la aplicación. Las principales etiquetas son las siguientes.
    - *Build:* atributo que contiene las plataformas de cuales se va a generar la aplicación.
    - Orientación: atributo que contiene el tipo de orientación de la aplicación se cuentan con dos opciones *Portrait* y *Landscape*.
    - *Template:* atributos que especifican la posición del componente en la aplicación.
- *navigation*: un conjunto de atributos que permite almacenar las características de navegación de la aplicación desde un inicio hasta un fin. Las principales etiquetas son las siguientes.
    - Id: atributo que hace referencia a un identificador único del componente
    - *Name*: atributo que especifica el nombre del componente
    - *Title:* atributo que representa el nombre desplegado del componente
    - *Type:* atributo que determina si el patrón es simple o compuesto.
    - *Next:* atributo que especifica el orden de navegación a una página siguiente.
    - *Position*: atributo que especifica el orden del componente que en la interfaz gráfica.

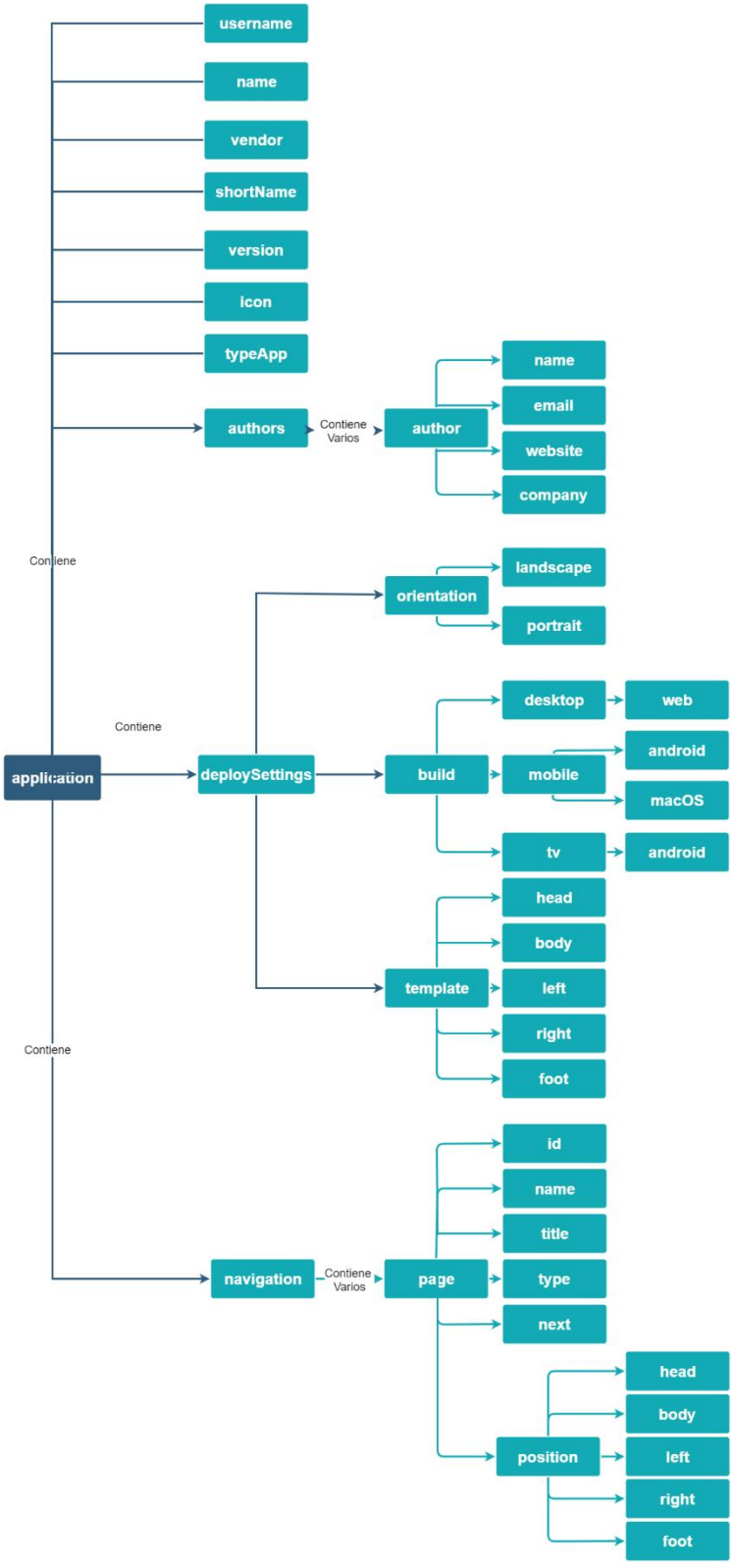


Fig. 3.28 Estructura del generador XML

3.6.2. Estructura final de los proyectos generados

En este punto se hace una breve descripción del contenido base de las plataformas que se necesita para generar un proyecto, es por eso que la estructura va determinada de acuerdo a las siguientes características (ver Fig. 3.29). ProjectBase es la carpeta principal y se divide en diferentes plataformas como: *Desktop*, *Mobile* y *TV*. *Desktop* cuenta con la carpeta Web, *Mobile* cuenta con la estructura de Android™, MacOS® y Web, y *TV* con Android™.

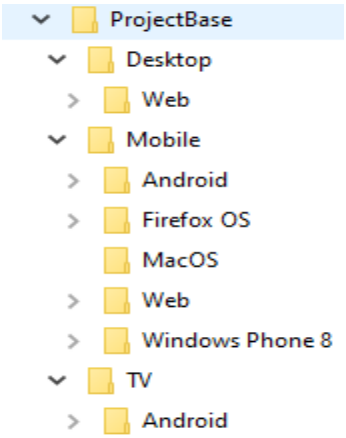


Fig. 3.29 Estructura de la carpeta ProjectBase

Dicho lo anterior, cada carpeta corresponde a un proyecto nuevo, estandarizado para ejecutar un proyecto básico que permita la manipulación de este. El proyecto tiene la característica de mantener una estructura de tal forma que el usuario lo exporte a un IDE de acuerdo a la plataforma de desarrollo, esto es, 1) los proyectos en Web usados desde cualquier tipo IDE con soporte para el desarrollo Web, 2) los proyectos en Android™ cuenta con soporte para el despliegue en Android™ Studio y 3) los proyectos creados para macOS® cuenta con soporte para el despliegue en Xcode.

La estructura del proyecto base para el desarrollo de aplicaciones Web, son para las siguientes plataformas *Desktop*, *Mobile* y *TV*. La estructura raíz de la carpeta se observa en la Fig. 3.30.

Fig. 3.30 Estructura de la carpeta para el proyecto Web

La estructura del proyecto base para el desarrollo de aplicaciones en Android™, son para las siguientes plataformas *Mobile* y *TV*. La estructura de la carpeta se observa en la Fig. 3.31.

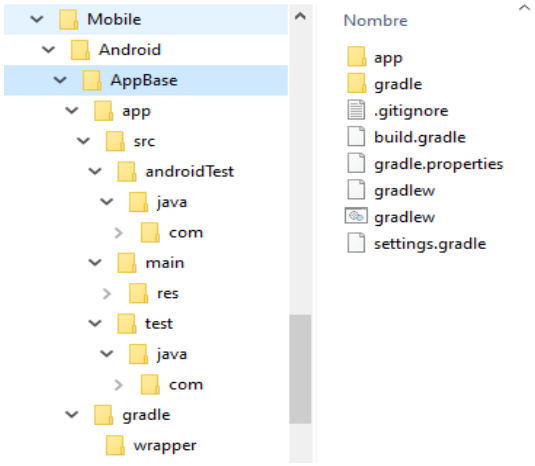


Fig. 3.31 Estructura de la carpeta para el proyecto Android™

Por otra parte, la estructura del proyecto base para el desarrollo de aplicaciones en macOS™, es para la plataforma *Mobile*. La estructura raíz de la carpeta se observa en la Fig. 3.32.

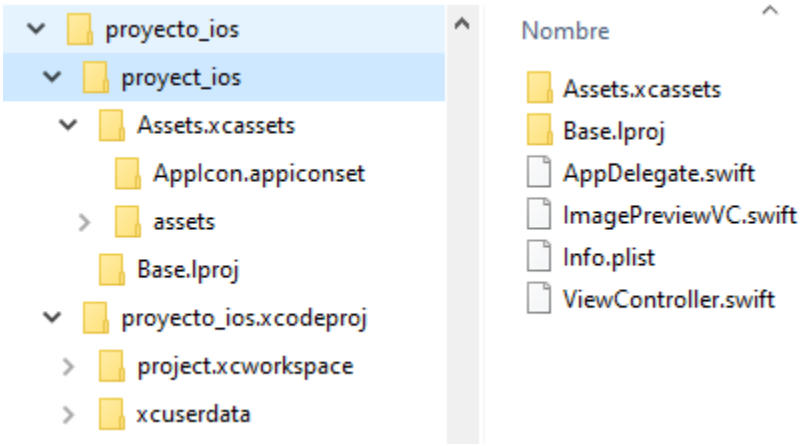


Fig. 3.32 Estructura de la carpeta para el proyecto macOS®

3.6.3. Servicio REST del módulo generador de aplicaciones

En esta subsección se describen las partes fundamentales de la ejecución del servicio REST, que genera de acuerdo a las características del archivo de configuración XML el código fuente de una aplicación.

En el Listado de código.3.5 la petición al módulo generador de aplicaciones, se realiza mediante los comandos *curl*, que permite verificar la conectividad URL, transferir el archivo XML y ejecutar el método solicitado. 1) en la línea 1 del código se invoca el método `writeFile ()` que permite escribir la estructura completa del XML. 2) En la línea 3 y 4 se especifica la dirección del servidor donde se encuentra el servicio, 3) en la línea 5 se inicializa la conexión con el servicio, 4) en la línea 6 se crea un arreglo que envía los siguientes parámetros que el servicio solicita para la correcta generación de la aplicación. Los parámetros se describen a continuación.

- *projectNameString*: variable del nombre proyecto
- *atilaString*: variable que contiene el nombre del archivo XML
- *type*: nombre del método que permite generar la aplicación basado en las características del archivo XML.

5) En la línea 13, 14 y 15 se ejecuta el comando CURL para enviar y recibir la información generada por el servicio.

Listado de código. 3.6 Servicio.php

```
1.  $this->writeFile($xml);
2.  //var_dump($xml);
3.  $actual_link = "http://$_SERVER[HTTP_HOST]";
4.  $service_url = $actual_link."/Atila-Desarrollo/entities/uploadFiles.php";
5.  $curl = curl_init($service_url);
6.  $curl_post_data = array(
7.  "projectNameString" => $this->appname,
8.  "atilaString" => $xml,
9.  "type" => "upstring" );
10. curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
11. curl_setopt($curl, CURLOPT_POST, true);
12. curl_setopt($curl, CURLOPT_POSTFIELDS, $curl_post_data);
13. $curl_response = curl_exec($curl);
14. curl_close($curl);
15. return $curl_response;
```

El servicio retorna un archivo ZIP con la estructura de carpetas de los proyectos generados por plataforma y por dispositivos especificados en el documento de configuración XML.

Capítulo 4 Resultados

En este capítulo se exponen dos diferentes casos, estas pruebas tiene como objetivo generar aplicaciones multidispositivo de tipo *e-Commerce* y *Social Media* por medio de una imagen que representa una interfaz para uno de estos dos tipos de aplicaciones. El proceso de identificación de elementos es generado por una red neuronal convolucional y a partir de los resultados obtenidos se realiza la generación automática del código fuente de la aplicación. Este proceso de generación de software consiste en un conjunto de pasos a seguir para cada caso de estudio.

El primer caso de estudio consiste en ingresar una imagen que representa una composición de UIDPs no ideal y agregar los datos de configuración para generar una aplicación de tipo *e-Commerce* para un dispositivo móvil Android™. El segundo caso de estudio plantea la generación de una aplicación *Social Media*, de forma similar al primer caso de estudio se parte de una imagen real a color que representa una composición de UIDPs con datos de configuración para un dispositivo Android™ TV.

4.1. Caso de estudio 1: Generación de una aplicación de *e-Commerce*

Para el primer caso de estudio, se propone la generación de un proyecto de tipo *e-Commerce* para un dispositivo móvil en Android™, se presenta el funcionamiento del generador de aplicaciones, que implican la carga de la imagen, selección de patrones, selección de dominio, selección de plantilla, selección del tipo de aplicación, selección del tipo de plataforma y dispositivo, datos de configuración general y la descarga del código fuente generado. Por último, la aplicación se despliega en el IDE con el objetivo de revisar el funcionamiento.

4.1.1. Diseño de la aplicación a generar

En esta subsección se explican los componentes de diseño para la aplicación a realizar. En este sentido, el caso de estudio está constituido por la composición que lleva por nombre Bus-men-sho-pro-com, orientado para la compra de uno a varios productos con un formato B2C, los UIDPs que la integran son los siguientes:

- *Search*: localizado en el parte superior derecho permite buscar productos que se encuentra almacenados.
- *Menu*: Localizado en el parte superior izquierdo permite acceder a otras vistas que motivan al usuario para visitar un producto de su interés, comprarlo o comentarlo y facilitar la navegación en la aplicación.
- *Product* Page: Permite ver las características del producto seleccionado, que incluye una imagen del producto y la información de venta.
- *Shopping Cart:* Permite al usuario comprobar los productos seleccionados por el usuario y ver el total de los productos seleccionados por el usuario.
- *Comment:* Permite agregar descripciones u opiniones de los usuarios acerca del producto que se desea comprar.

En la figura 4.1 se observa el diseño de esta composición.

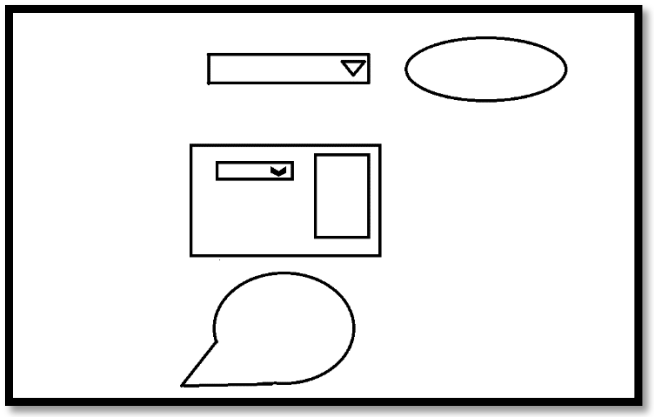


Fig. 4.1 UIDP para el caso de estudio 1

Una vez descrita la composición de UIDPs, el siguiente paso es generar una imagen no ideal, la imagen cuenta con una resolución de 1050 * 650 pixeles y generado en un editor de dibujo que se tomó en cuenta el diseño de la Fig. 4.1 tal como se observa en la Fig. 4.2.

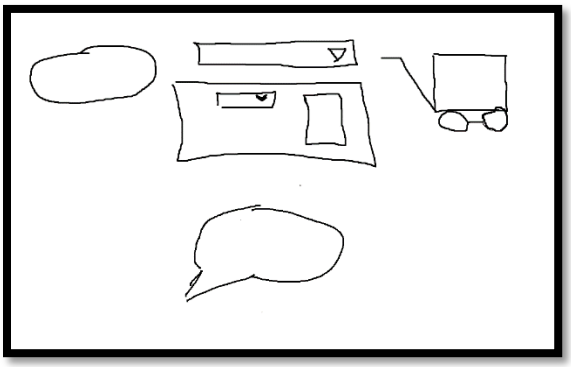


Fig. 4.2 UIDP no ideal para el caso de estudio 1

El UIDP dibujado a mano alzada se utilizó en la herramienta para la generación de la aplicación.

4.1.2. Generación de aplicación

En esta subsección se continúa con el caso de estudio, en esta parte se describe el funcionamiento de la aplicación para generar un proyecto conforme a la imagen se ingresa y se agrega las características al proyecto.

El primer paso es ingresar a la pestaña *Freehand*, donde se muestra el formulario para seleccionar la imagen y el botón de *Upload File* para iniciar el proceso de identificación (ver Fig. 4.3).

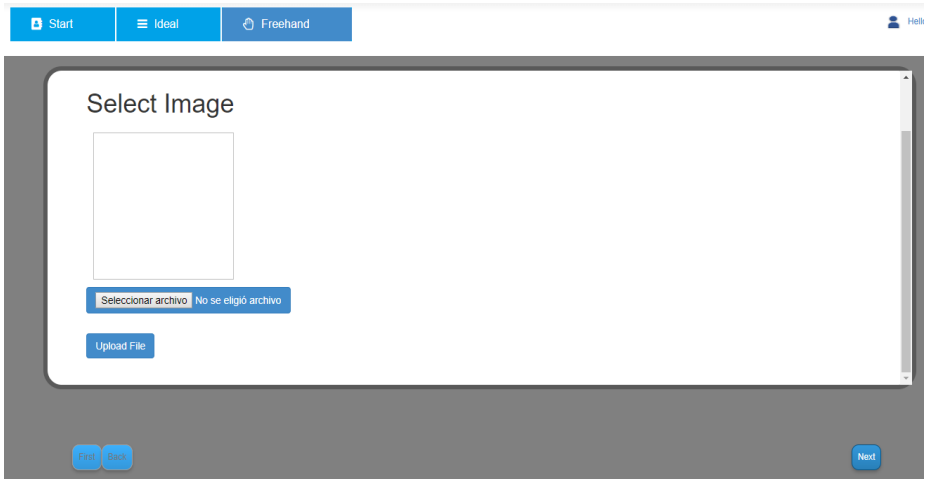


Fig. 4.3 Menú del generador pestaña Freehand

Al darle al botón seleccionar archivo abrirá una ventana de selección y se buscará la imagen que representa la composición de UIDPs (ver Fig. 4.4).

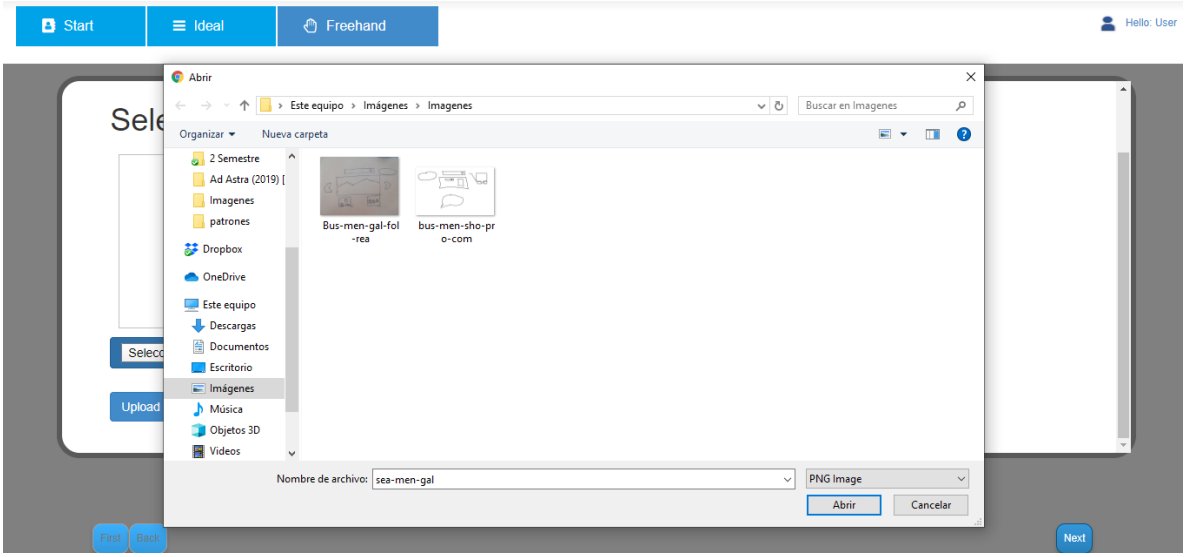


Fig. 4.4 Pestaña Freehand selección de imagen

Al darle al botón *Abrir* pre-visualiza la imagen cargada en el seleccionador de imagen. Al darle clic al botón *Upload File* continúa con el siguiente paso (ver Fig. 4.5).

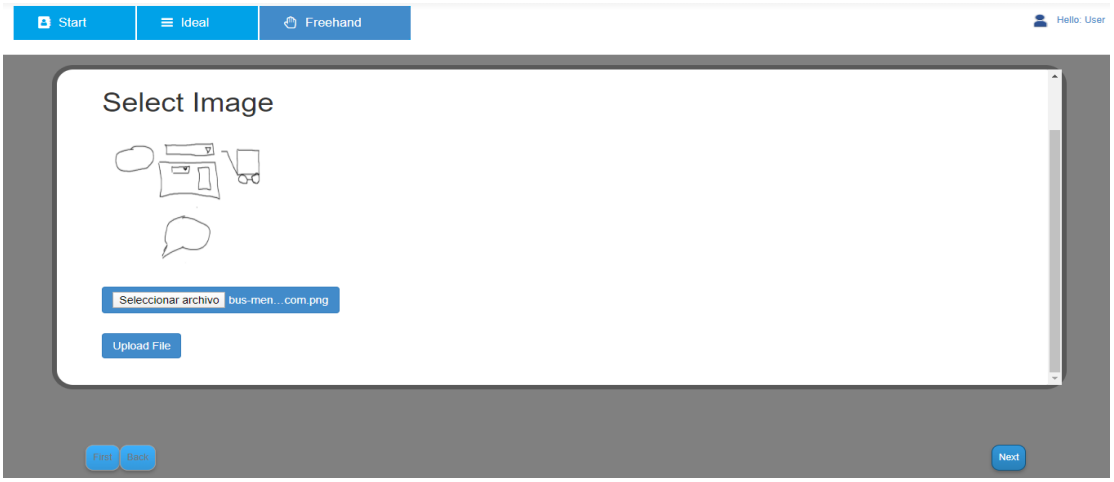


Fig. 4.5 Pestaña Freehand selección de imagen

Una vez dado clic en *Upload File* hará una carga de la imagen y hará un tiempo de espera con un promedio de 5 segundos para continuar (ver Fig. 4.6).

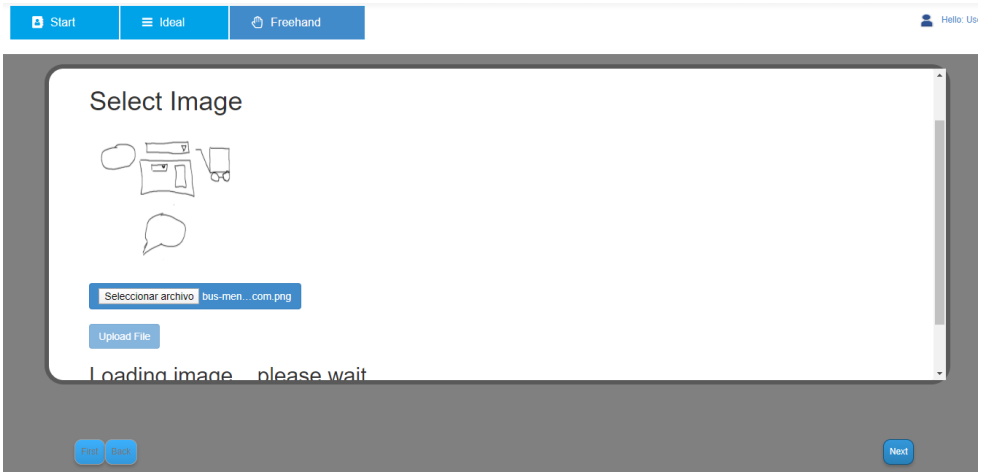


Fig. 4.6 Pestaña Freehand subiendo archivo

Al terminar al cargar la imagen, el mensaje *Loading imagen* desaparecerá, a través de un modal se notifica e indica la clasificación de los cuatro primeros UIDPs identificados que tengan parecido, para el modal se tendrá dos opciones: *Change UIDPs* que permite seleccionar los UIDPs clasificados y *I agree UIDPs que* permite pasar al proceso de selección de dominio, omitiendo la elección de UIDPs (ver Fig. 4.7).

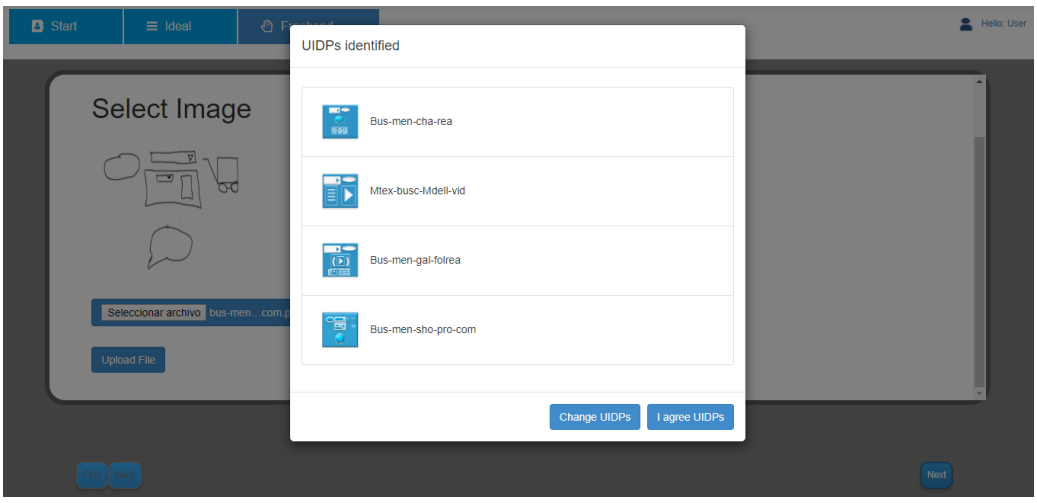


Fig. 4.7 Pestaña Freehand UIDPs identificados

En este caso, se elige la opción *Change UIDPs*, la pantalla de selección de patrones desplegará los cuatro primeros patrones identificados y seleccionados, (ver Fig. 4.8).

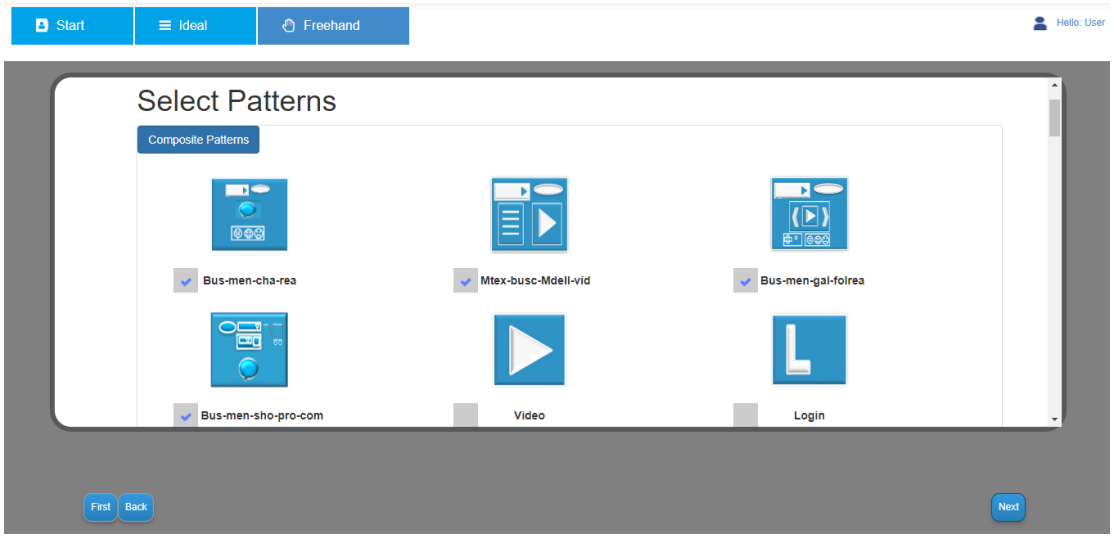


Fig. 4.8 Pestaña Freehand Select Patterns

Por último, se elegirá un solo patrón de composición que más se parezca al UIDP original, el cual es patrón Bus-men-sho-pro-com (ver Fig. 4.9).

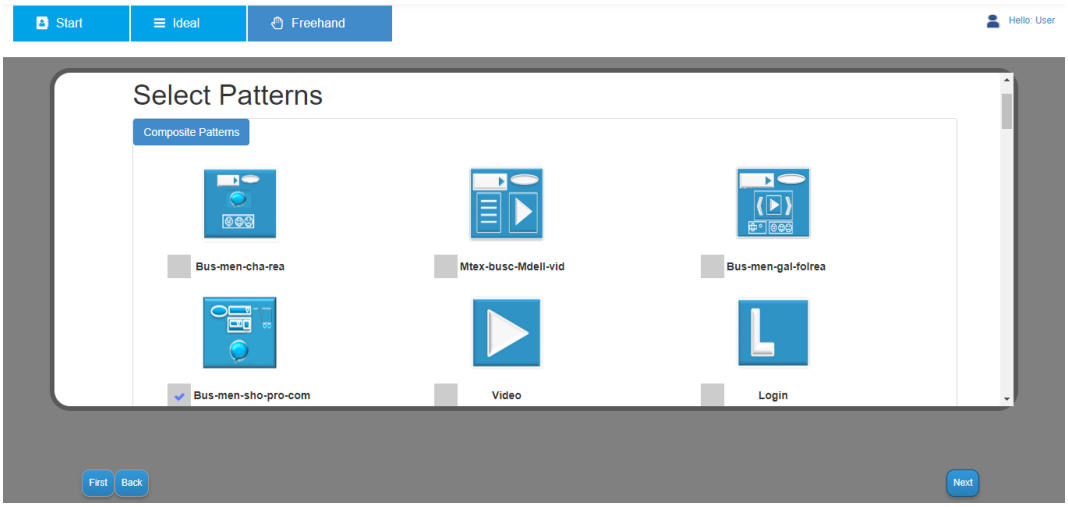


Fig. 4.9 Pestaña Freehand Select Patterns

Al continuar con el proceso se muestra la sección de Dominio, para este patrón su relación corresponde solamente para el dominio entretenimiento, por lo que se selecciona y continúa con el proceso (ver Fig. 4.10).

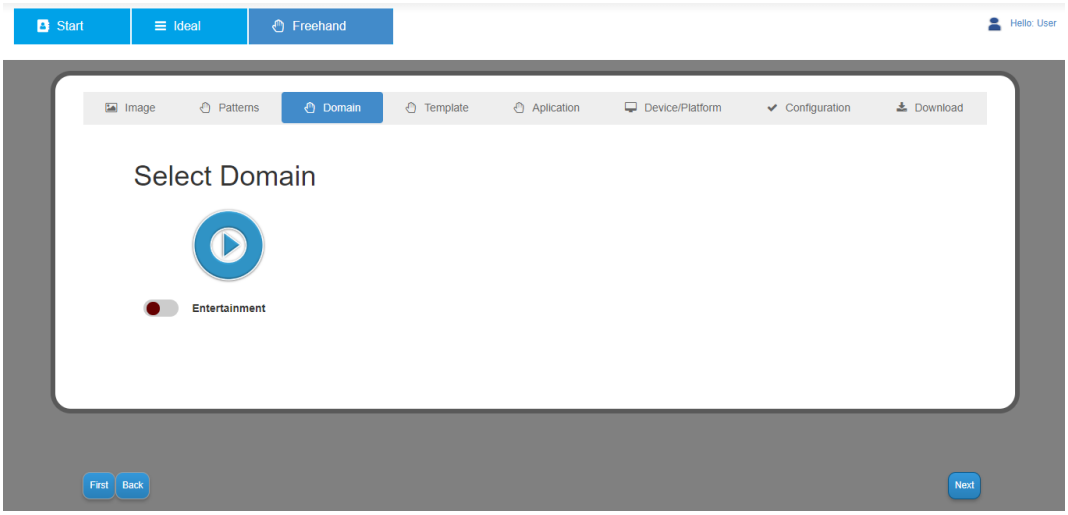


Fig. 4.10 Pestaña Freehand Select Domain

El siguiente paso es la selección de plantillas, para este caso se cuenta con una relación ya que no todas las plantillas son aplicables para el patrón identificado (ver Fig 4.11)

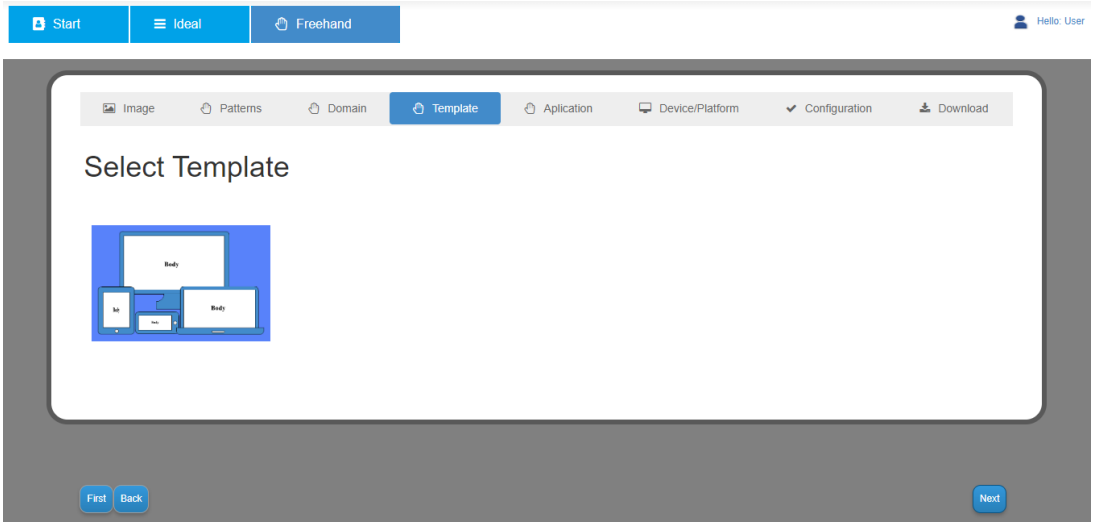


Fig. 4.11 Pestaña Freehand Select Template

Ahora se selecciona el tipo de aplicación e-Commerce, en este caso se cuenta con B2C y C2C, este caso de estudio se trata de una aplicación B2C (ver Fig. 4.12).

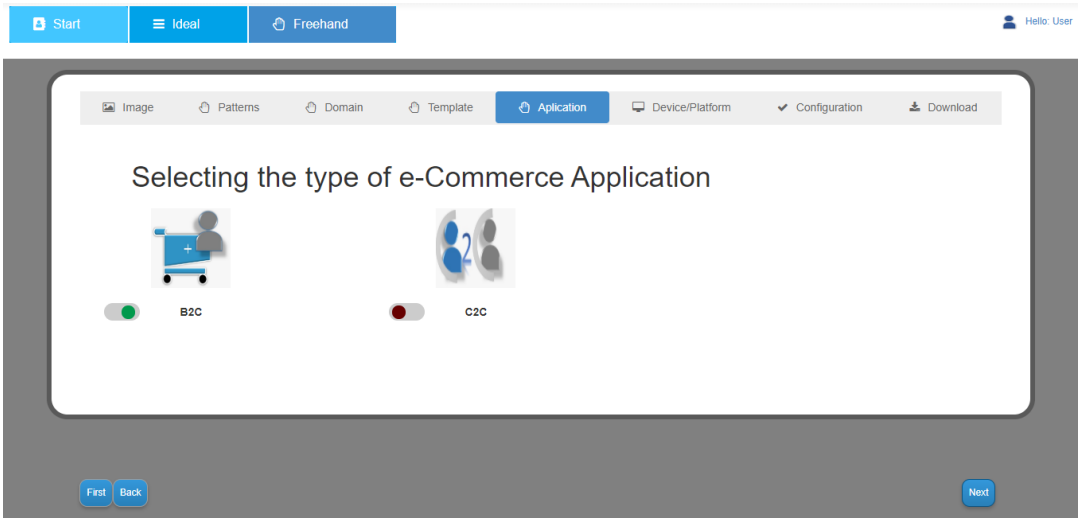


Fig. 4.12 Pestaña Freehand Select the type of Entertainment Application

Ahora, la selección de plataforma y dispositivos, que para este caso se elige la opción Tablet/Smartphone y se selecciona la opción para la versión de Android™ (ver Fig. 4.13).

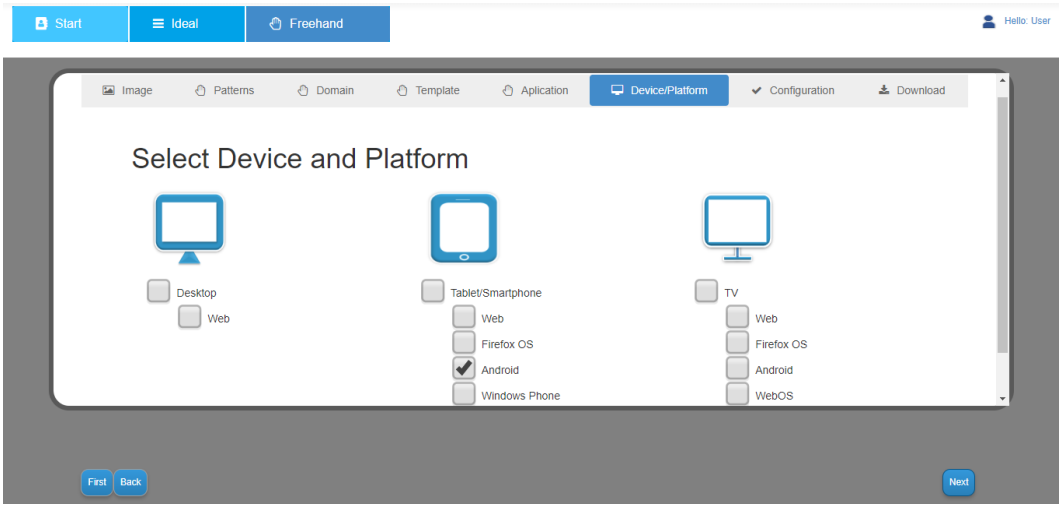


Fig. 4.13 Pestaña Freehand Select Device and Platform

Al continuar, en la pantalla de configuración permite agregar las propiedades al proyecto, se llena un pequeño formulario con datos generales de la aplicación y datos del autor de la aplicación. Para la aplicación se cuenta con el título de aplicación, un nombre corto y la versión que requiere. Mientras que para el autor se cuenta con el nombre, la compañía, el email y el sitio Web (ver Fig. 4.14).

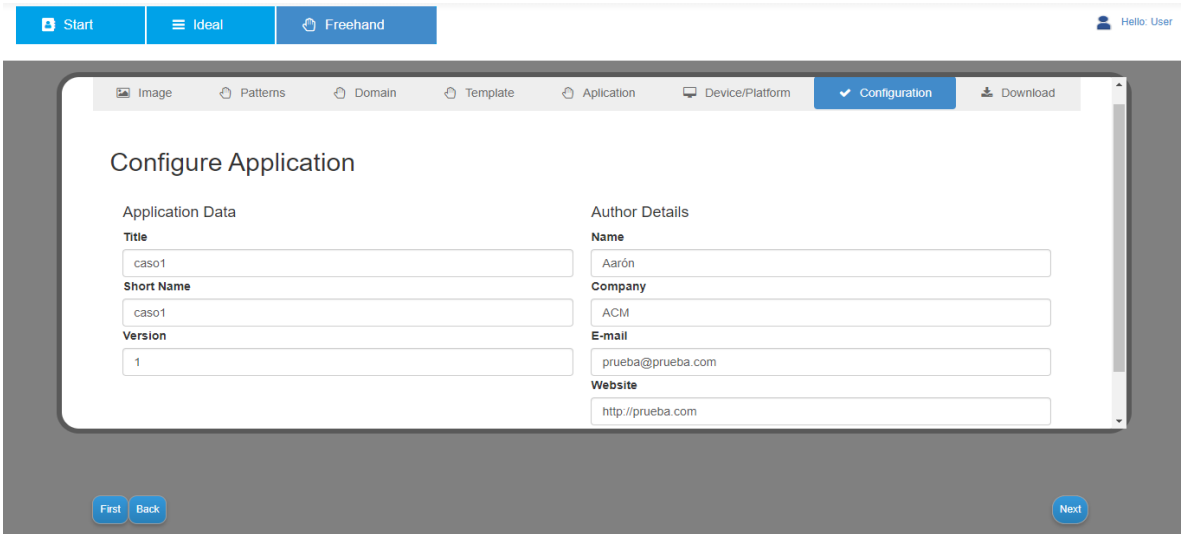


Fig. 4.14 Pestaña Freehand Configure Application

Al enviar los datos, la herramienta solicita validar la información a través de un modal con un breve resumen de los datos de configuración que cuenta el proyecto a generar. Se muestra la siguiente información: el nombre de la aplicación, el dominio, el tipo de patrón seleccionado, la plantilla y el dispositivo seleccionado. En caso de aceptar, cargará la pestaña Descarga (ver Fig. 4.15). En caso de que se cancele, permite cambiar la configuración de las propiedades.

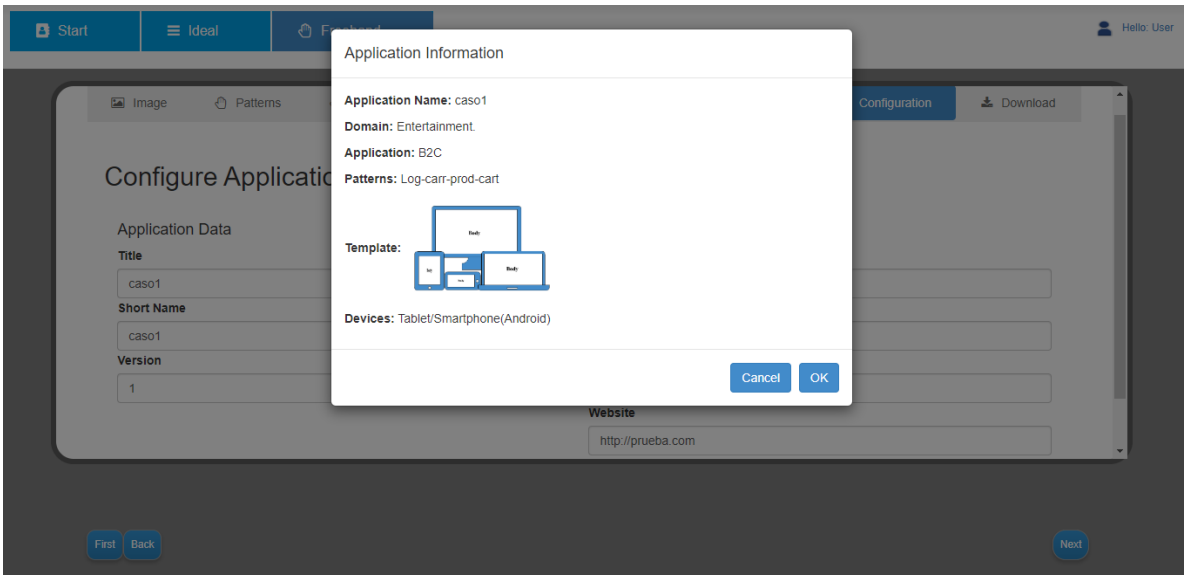


Fig. 4.15 Pestaña Freehand Configure Application

Al terminar con los anteriores pasos, se envía a la descarga del proyecto, mediante un archivo ZIP y se descargará automáticamente o en forma manual (ver Fig. 4.16).

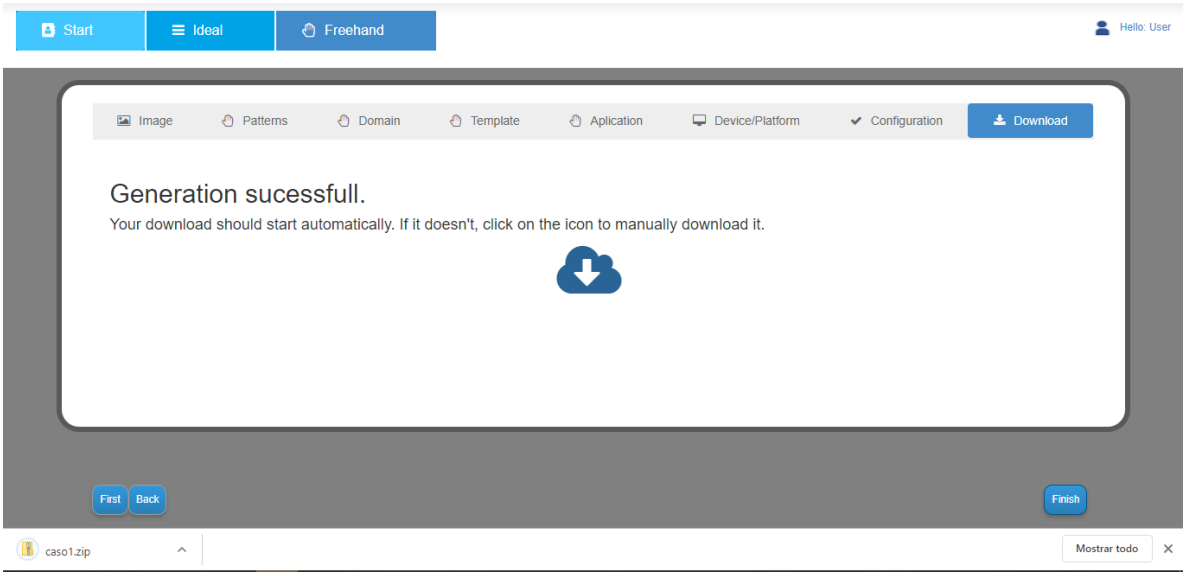


Fig. 4.16 Pestaña Freehand Download

Como último paso, se descomprime y se revisa el proyecto generado por la herramienta. (ver Fig. 4.17).

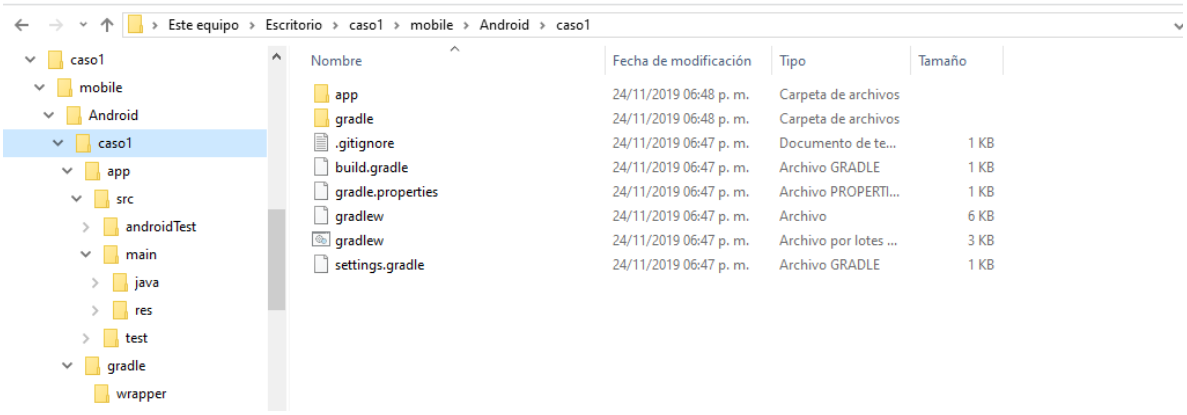


Fig. 4.17 Archivo zip

4.1.3. Revisión de la aplicación generada

Para culminar el caso de estudio, se revisa la aplicación en el entorno de desarrollo para verificar que las clases y la configuración del proyecto se encuentren de forma adecuada y listo para ejecutar. A continuación, se muestran los patrones generados (ver fig. 4.18).

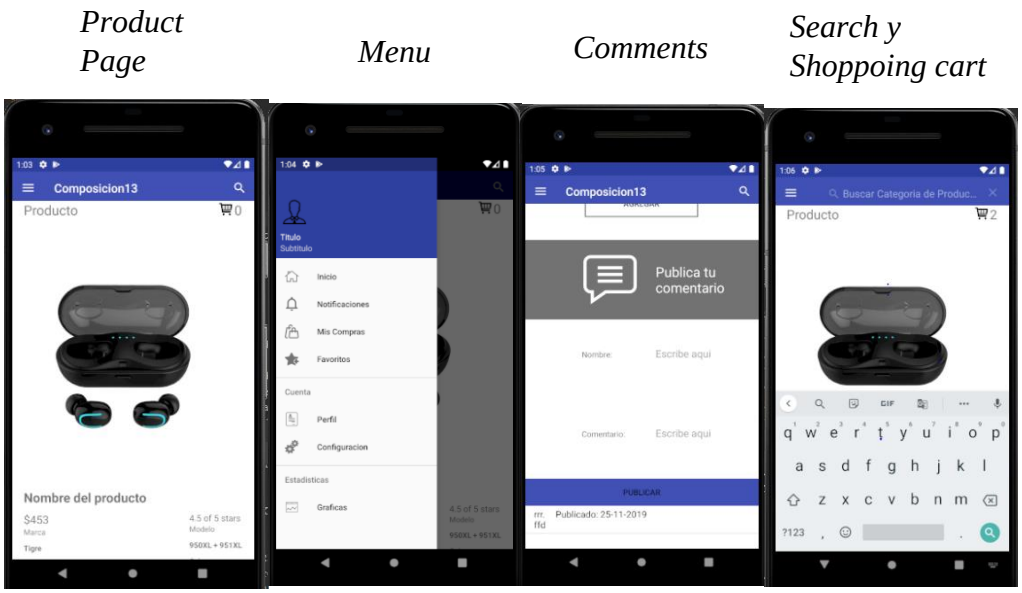


Fig. 4.18 Revisión del Proyecto del caso de estudio 1

Como conclusión se menciona que el proyecto fue totalmente generado, y listo para su uso. En este caso se generó una aplicación e-Commerce para agregar productos a la lista de compras, escribir y revisar comentarios del producto, buscar productos y revisar las estadísticas del producto.

4.2. Caso de estudio 2: Generación de una aplicación Social Media

Para el segundo caso de estudio, se propone la generación de un proyecto de tipo Social Media para un dispositivo Smart TV basado en Android.

4.2.1. Diseño de la aplicación a generar

En esta subsección se explican los componentes de diseño para la aplicación a realizar. En este sentido, el caso de estudio está constituido por la composición que lleva por nombre Bus-men-gal-fol-rea, orientado a una aplicación de tipo red social cuyas características permite a los usuarios revisar imágenes de la comunidad, buscar imágenes, reaccionar a una imagen y ver las últimas imágenes actualizadas por el usuario, los UIDPs requeridos para esta aplicación son los siguientes:

- *Search*: Localizado en la parte superior derecha, permite buscar imágenes de los usuarios.
- *Menu*: Localizado en la parte superior izquierda, permite acceder a otras vistas que motivan al usuario a visitar otras secciones de su interés, tales como revisar imágenes, calificar y seguir a más usuarios.
- *Gallery*: Permite ver las imágenes de los usuarios y de la comunidad.
- *Follow*: Permite seguir a usuarios y suscribirse a las actualizaciones de sus últimas imágenes publicadas.
- *Reaction*: Permite reaccionar a las imágenes de los usuarios.

En la Fig. 4.19 se observa el diseño de esta composición.

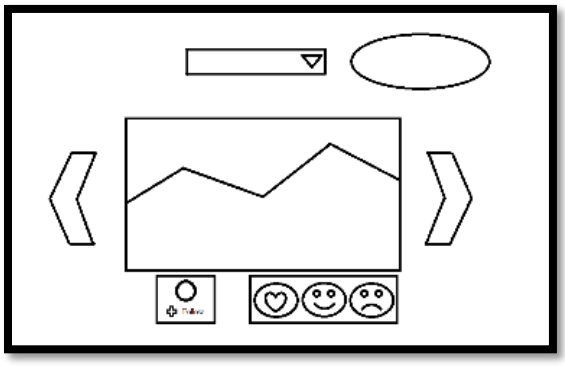


Fig. 4.19 UIDP para el caso de estudio 2

Una vez descrita la composición de UIDPs, el siguiente paso es generar una imagen real a color con dimensión de 1050 * 650 pixeles con formato PNG, el dibujo realizado en una hoja de color amarilla y trazado con un lapicero de color azul, la fotografía tomada con un dispositivo móvil con un lente de 10 megapíxeles, se tomó en cuenta el diseño de la Fig. 4.19 tal como se observa en la Fig. 4.20.

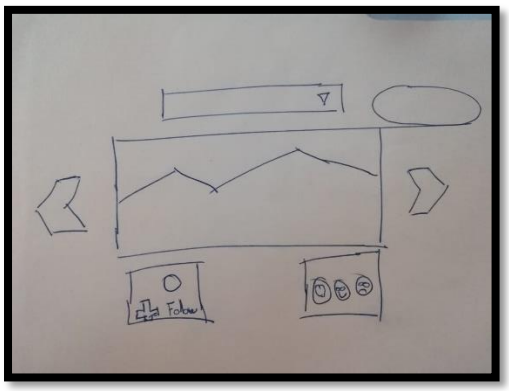


Fig. 4.20 UIDP no ideal para el caso de estudio 2

La composición dibujada a mano alzada se agrega en la herramienta para la generación de la aplicación.

4.2.2. Generar aplicación

En esta subsección se continúa con el caso de estudio, en esta parte se describe el funcionamiento de la aplicación para generar un proyecto conforme a la imagen ingresada y agregar las características al proyecto.

Siguiendo los pasos del anterior caso de estudio, como primer paso se ingresa a la pestaña *Freehand*, donde se muestra el formulario para seleccionar la imagen y el botón de *Upload File* para iniciar el proceso de identificación (ver Fig. 4.21).

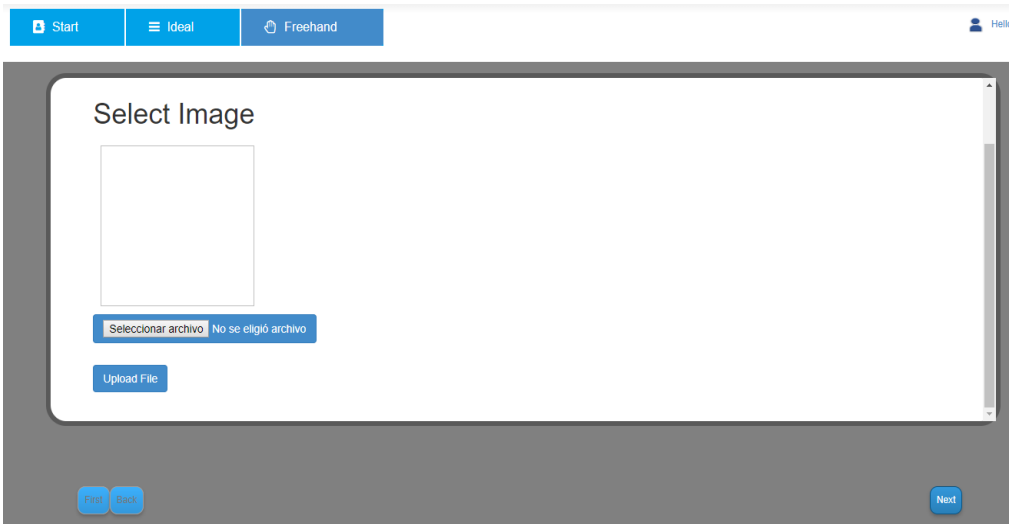


Fig. 4.21 Menú del generador pestaña Freehand

Al darle al botón seleccionar archivo nos abrirá una ventana de selección y para buscar la imagen que representa la composición de UIDPs (ver Fig. 4.22).

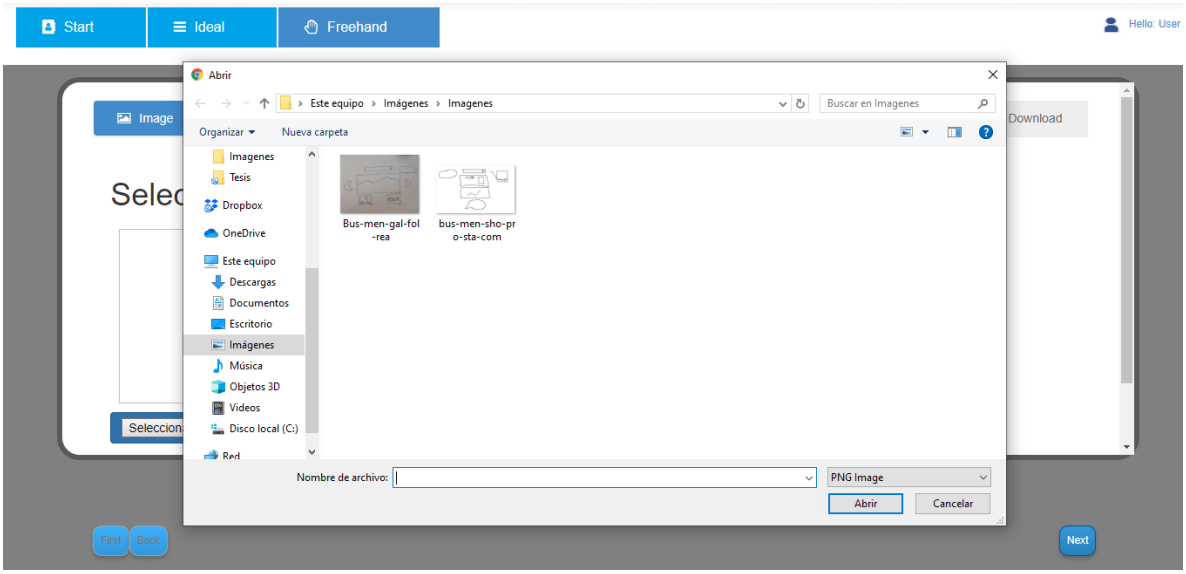


Fig. 4.22 Pestaña Freehand selección de imagen

Una vez seleccionada la imagen, se presiona el botón *Upload File* (ver Fig. 4.23).

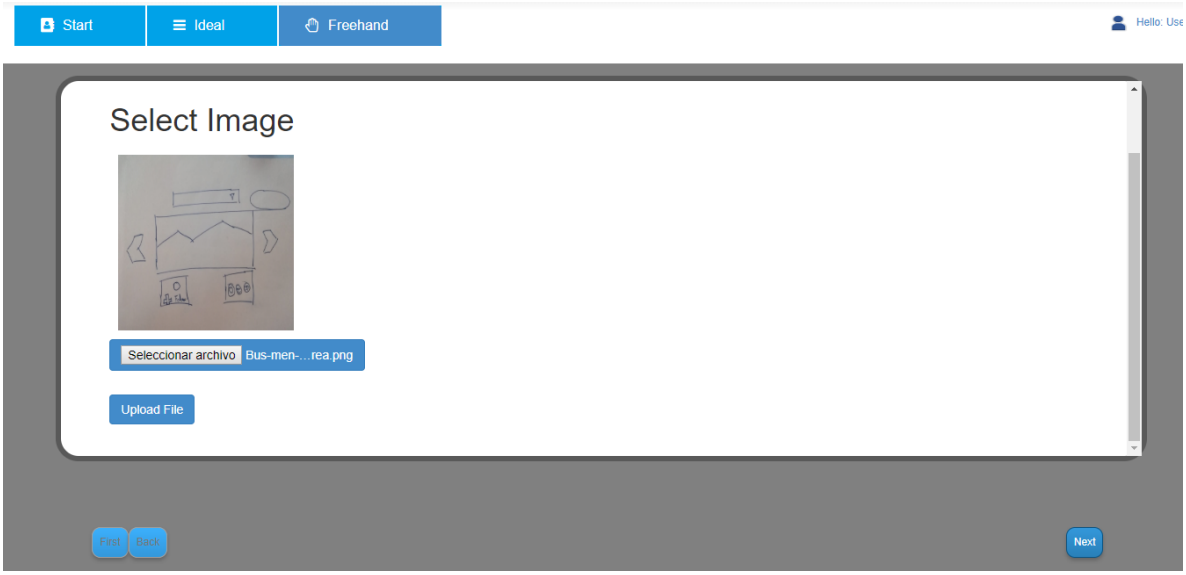


Fig. 4.23 Pestaña Freehand selección de imagen

Al continuar con la selección de la imagen, se carga la imagen (ver Fig. 4.24).

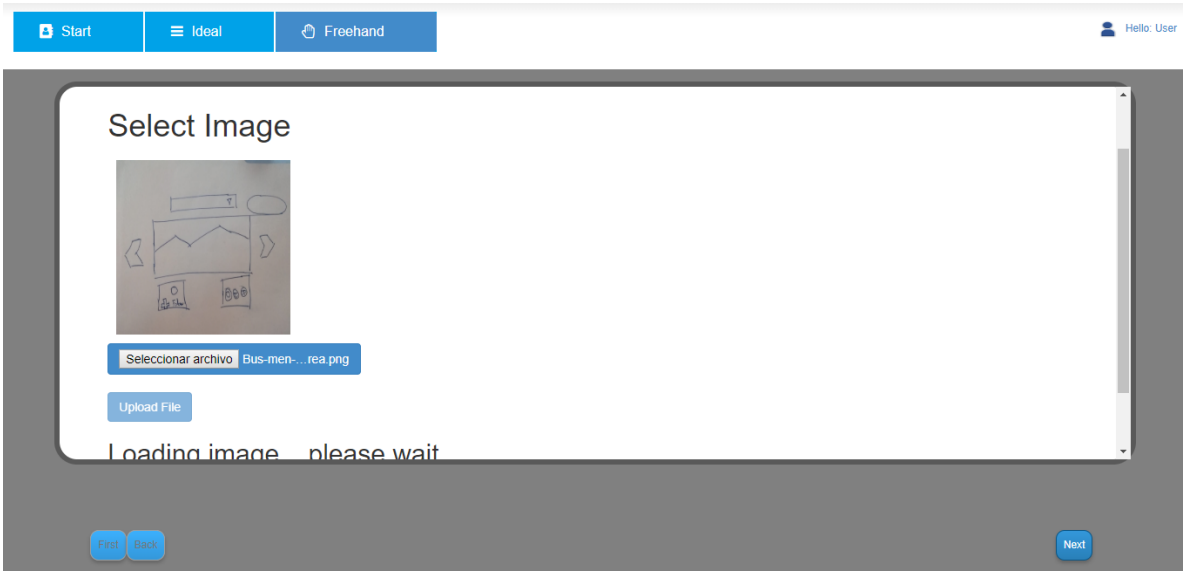


Fig. 4.24 Pestaña Freehand subiendo archivo

Ahora, el modal notifica e indica la clasificación de los cuatro primeros UIDPs identificados, estos son: bus-tag-pag, busc-mdell-vid, busc-carr y bus-me-gal-fol-rea (ver Fig. 4.25).

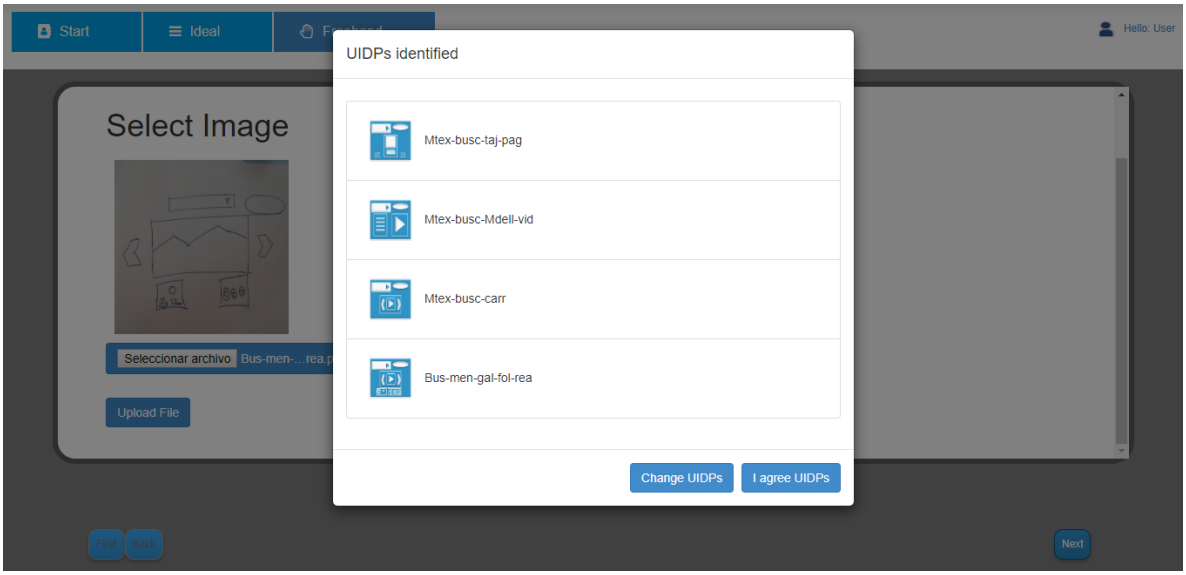


Fig. 4.25 Pestaña Freehand UIDPs identificados

La pantalla de selección de patrones desplegará los cuatro primeros patrones identificados y seleccionados, (ver Fig. 4.26).

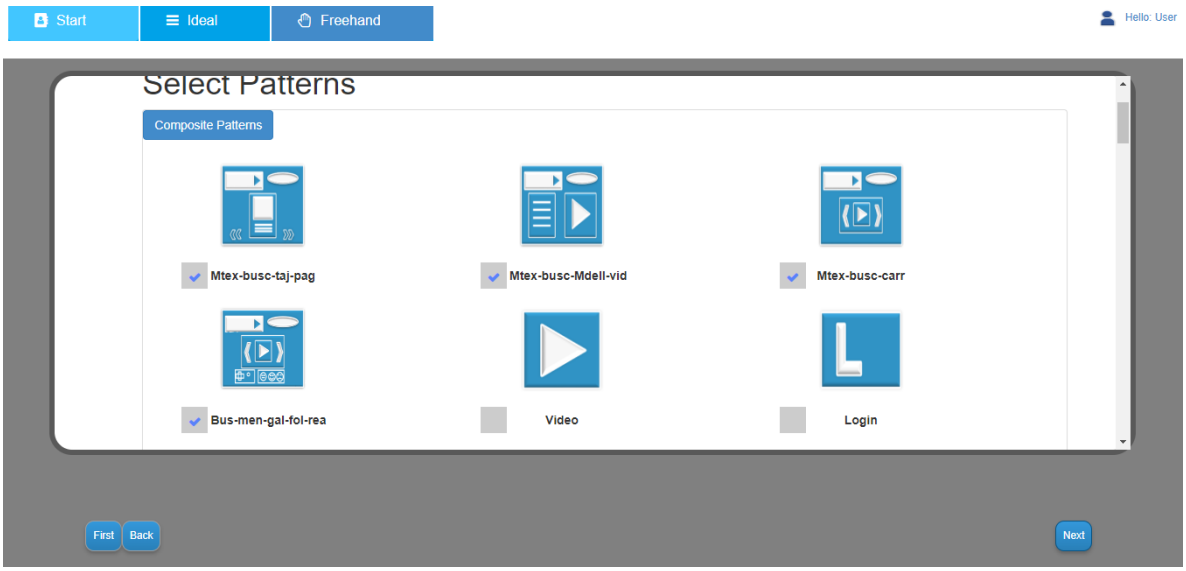


Fig. 4.26 Pestaña Freehand Select Patterns

Se elige un solo patrón de composición que más se parezca al UIDP original, el cual es la composición: Bus-men-sho-pro-sta-com (ver Fig. 4.27).

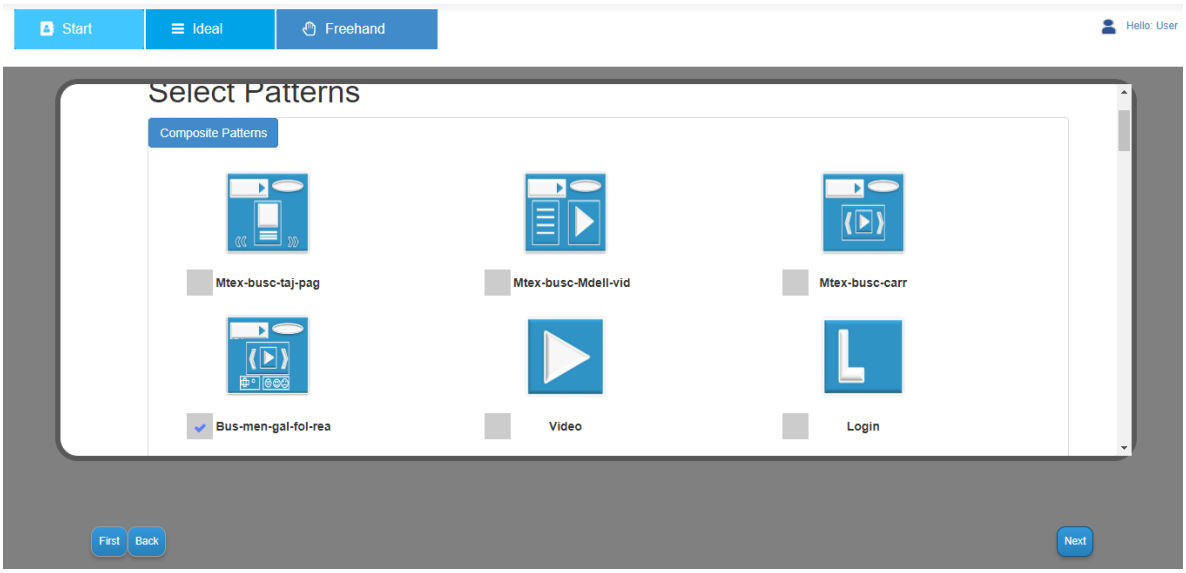


Fig. 4.27 Pestaña Freehand Select Patterns

Al continuar con el proceso, se muestra la sección de Dominio, para este patrón su relación corresponde solamente para el dominio entretenimiento (ver Fig. 4.28).

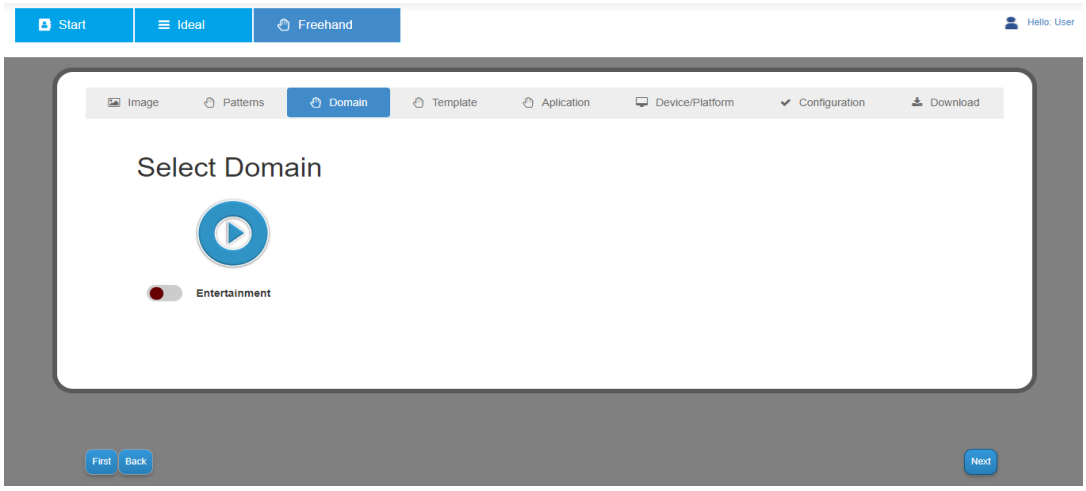


Fig. 4.28 Pestaña Freehand Select Domain

En la imagen aparece una sola plantilla a seleccionar la opción visto en el caso anterior (ver Fig 4.29)

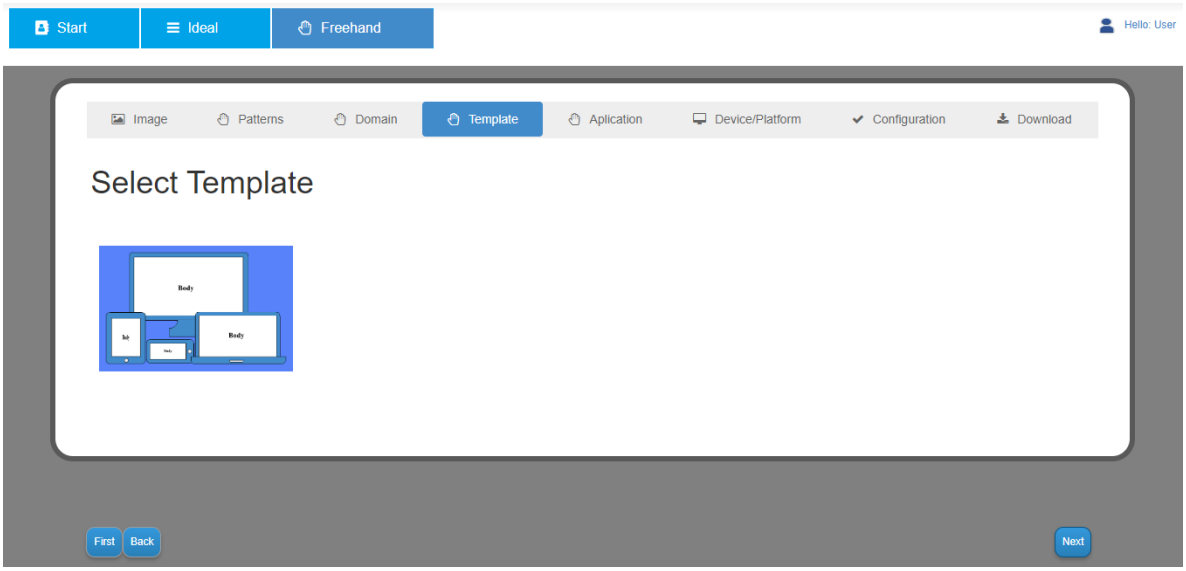


Fig. 4.29 Pestaña Freehand Select Template

Ahora se selecciona el tipo de aplicación Social Media, que en este caso es Redes sociales. (ver Fig. 4.14).

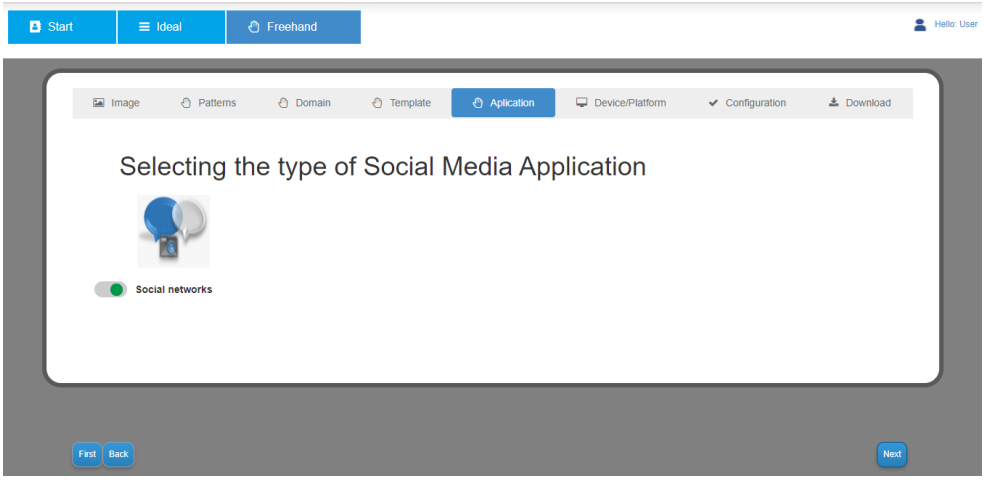


Fig. 4.30 Pestaña Freehand Select the type of Entertainment Application

En seguida, se cuenta con la selección de plataforma y dispositivos, para este caso se elige la opción Smart Tv y se selecciona la opción para la versión de Android™ (ver Fig. 4.31).

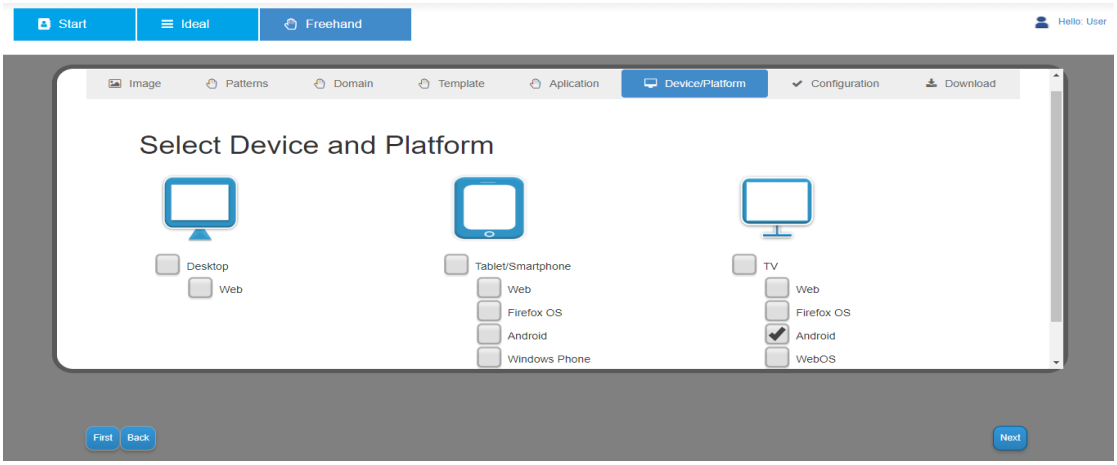


Fig. 4.31 Pestaña Freehand Select Device and Platform

Al continuar, en la pantalla de configuración se agregan las propiedades al proyecto, se rellena un formulario con datos de despliegue de la aplicación, incluyendo datos del autor (ver Fig. 4.32).

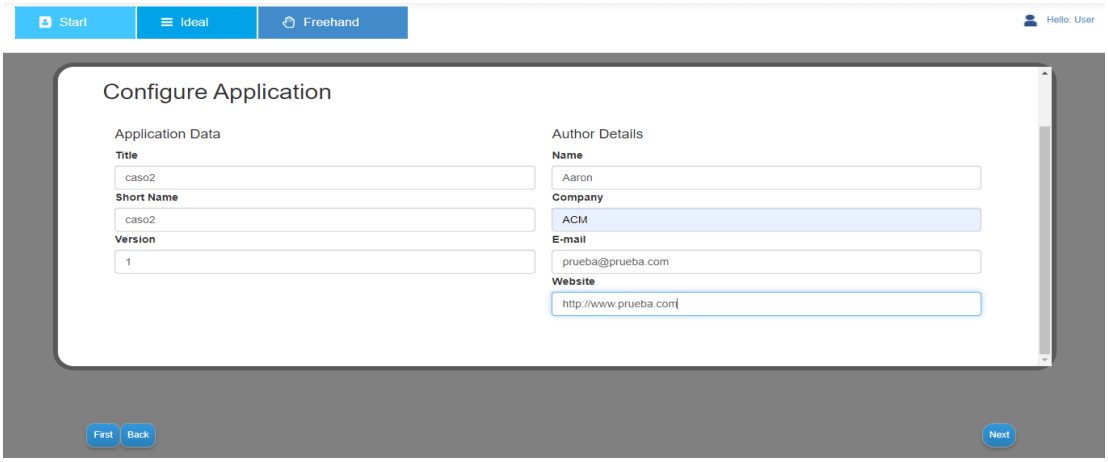


Fig. 4.32 Pestaña Freehand Configure Application

Pasa a la validación, se abre un modal con un breve resumen la configuración del proyecto a generar (ver Fig. 4.33).

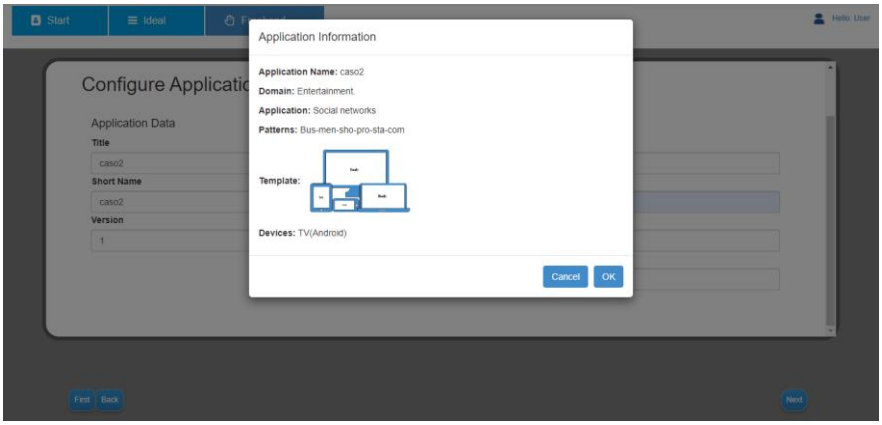


Fig. 4.33 Pestaña Freehand Configure Application

Para terminar, la herramienta envía a la pantalla de Descarga del código fuente generado mediante un archivo ZIP (ver Fig. 4.34).

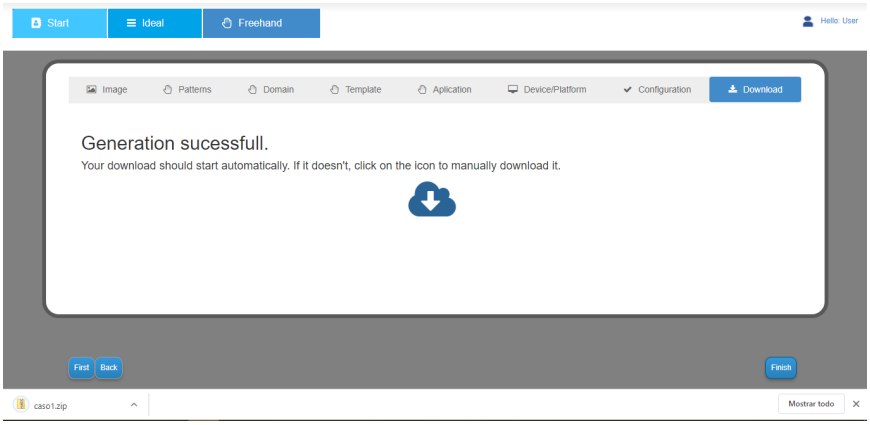


Fig. 4.34 Pestaña Freehand Download

Finalmente, se descomprime el archivo ZIP y se revisa el proyecto generado por la herramienta (ver Fig. 4.35).

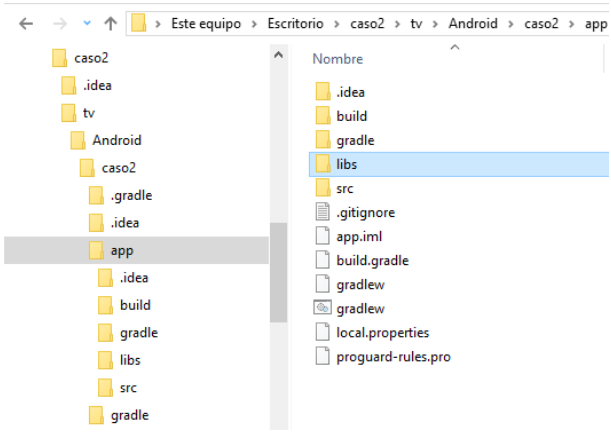
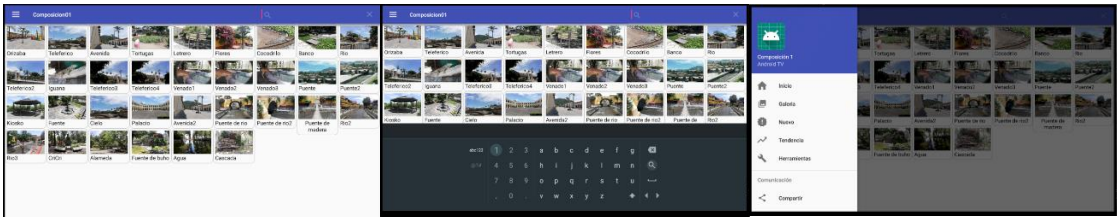


Fig. 4.35 Archivo zip

4.2.3. Revisión de la aplicación generada

Por último, la aplicación se prueba en el entorno de desarrollo para verificar las clases y la configuración del proyecto, los resultados se mostraron en la Fig. 4.36

Gallery *Search* *Menu*



Reaction y follow

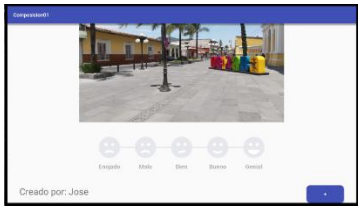


Fig. 4.36 Revisión del Proyecto Caso de estudio 2

Como conclusión, para el segundo caso de estudio se comprueba que la generación para dispositivos TV la cual está lista para su compilación y ejecución. Por lo tanto, generar una aplicación Social Media para las necesidades de los usuarios para dispositivos TV es satisfactoria y queda para el usuario la disponibilidad de ver, buscar calificar y seguir imágenes de uno o más usuarios y para el caso del desarrollador tiene un vínculo más sencillo de generar un proyecto satisfactorio para una rápida manipulación del código fuente generado.

Capítulo 5 Conclusión y recomendaciones

5.1. Conclusiones

Los resultados obtenidos gracias al apoyo de las técnicas de *Deep Learning* aportan un gran avance para estimular futuros desarrollos para diferentes ámbitos como aplicaciones médicas. Este aporte en la ingeniería de software, da un panorama amplio para los desarrolladores que buscan procedimientos exactos, confiables y con costo menor de lo promediado. Cabe aclarar que la integración del *Deep Learning*, conlleva a una nueva línea de investigación para desarrollar e implementar aplicaciones en diferentes dominios y crea nuevas técnicas, métodos, paradigmas y metodologías que ampliarán una expectativa diferente a las técnicas convencionales que aún se trabajan.

Los casos de estudio presentados demuestran un claro ejemplo de que las diferentes imágenes de prueba (imágenes ideales, no ideales, reales en blanco y negro, y reales a color), sin problema de clasificación. Los resultados obtenidos no solamente obtuvieron un impacto importante para la clasificación, sino también el tiempo de espera para reconocer un patrón se concluye que es adecuado.

Este trabajo es una extensión de investigaciones previas que se presentan en Cortes-Camarillo et al. [24], Cortes-Camarillo et al. [25], Sánchez-Morales et al. [26] y Sánchez-Morales et al. [28] cuyas pruebas y resultados son un margen de referencia para mejorar los algoritmos de identificación de UIDPs en imágenes generadas a mano alzada.

Gracias a estas nuevas técnicas de desarrollo de software, se logra administrar mejor los proyectos, factibilidad del proyecto y también la reducción del costo del desarrollo. Para los usuarios y los desarrolladores es importante siempre contar con herramientas que faciliten el desarrollo de software y permitan además una interacción mucho mayor con el entorno o ambiente que se manipula.

5.2. Recomendaciones

Al revisar los puntos observados, desarrollados, y generados del proceso de desarrollo de aplicaciones determinan los siguientes puntos especiales para atender las siguientes recomendaciones:

Es claro mencionar que este trabajo se realizó con los dominios de e-Commerce y Social Media, por tal motivo, el generar nuevos patrones compuestos que delimiten nuevas funcionalidades que aprovechen resolver algunos problemas específicos como integrar más patrones para hacer una aplicación más completa a la hora de generar una nueva aplicación, a la par, da la oportunidad de trabajar con otros dominios un ejemplo de ello es el dominio médico, ya que cuenta con otro tipo de patrones aún por conocer y analizar.

La utilización de la red neuronal convolucional, promueve la investigación de hallar nuevas formas de crear redes neuronales específicas para analizar y desarrollar e implementar nuevos modelos, para este caso un ejemplo sería es que se promueva un nuevo modelo que permite obtener características para la detección bordes más pronunciadas para el UIDP de *Page Product.* Esto es conveniente ya que solamente se adecuaría únicamente al entrenamiento y entendimiento de los patrones.

A pesar que se obtuvo un buen porcentaje del 74% en la precisión al clasificar los UIDPs aún falta mejorar esa precisión, sería conveniente aumentar categóricamente a más de 1000 instancias por patrón u ocupar una nueva versión de la red neuronal convolucional Inception_v4 que mejora ciertas capacidades de clasificar, como la optimización de la memoria al usar el algoritmo de propagación hacia atrás y la implementación de un nuevo diseño de agregar las versiones de la red neuronal de *Inception* en una sola.

Es necesario tener los componentes actualizados al día, por ello es necesario realizar seguimiento de la actualización y/o modificación de las versiones más recientes, esto con motivo de que no permita generar problemas con la compatibilidad de funcionalidad de los componentes porque tendría problemas de generación de la aplicación.

Productos académicos

En este apartado, se presentan los trabajos derivados de este proyecto de investigación.

Artículos

Aarón Colmenares Morales, Giner Alor-Hernández, Laura Nely Sánchez-Morales, Ulises Juárez Martínez, José Luis Sánchez-Cervantes. **Proceso de Desarrollo de aplicaciones para el dominio social media usando técnicas de Deep Learning**. Research in Computing Science Issue 148 (7) (2019). ISSN 1870-4069 **Estatus**: Publicado.

Estancia Académica

Universidad Veracruzana

Universidad Veracruzana. Xalapa, Ver. El Centro de investigación en Inteligencia (CIIA). Periodo: 1 de agosto al 31 de agosto del 2019. Dr. Héctor Gabriel Acosta Mesa y Dr. Nicandro Cruz Ramírez.

Referencias

[1] U. Schick. (2018, Marzo 29). *What is Artificial Intelligence?* [online]. Available: https://news.sap.com/2018/03/what-is-artificial-intelligence [Accessed: 18-Oct]

[2] (2018). *Deep Learning (Aprendizaje Profundo) Tres cosas que es necesario saber*. [online]. Available: https://es.mathworks.com/discovery/deep-learning.html [Accessed: 18-Oct]

[3] Y. LeCun et al., "Deep learning," *NATURE*, vol 521, pp. 436-444, May 2015.

[4] (2018). *Design Patterns*. [online]. Available: http://ui-patterns.com/patterns [Accessed: 18-Oct]

[5] M. Tsikerdekis and S. Zeadally, "Online Deception in Social Media," *Communications of the ACM*, vol 57, Issue 9, pp. 72-80, September 2014.

[6] (2018). *Comercio Electrónico*. [online]. Available https://azure.microsoft.com/es-mx/solutions/ecommerce/ [Accessed: 18-Oct]

[7] (2018). TensorFlow [online]. Available: https://www.tensorflow.org/?hl=es [Accessed: 18-Oct]

[8] (2018). *Theano at a Glance*. [online]. Available: http://deeplearning.net/software/theano/introduction.html [Accessed: 18-Oct]

[9] (2018). *Caffe*. [online]. Available: http://caffe.berkeleyvision.org/ [Accessed: 18-Oct]

[10] (2018). *Deep Learning for Java*. [online]. Available: https://deeplearning4j.org/ [Accessed: 18-Oct]

[11] (2018). *What is Python? Executive Summary*. [online]. Available: https://www.python.org/doc/essays/blurb/ [Accessed: 18-Oct]

[12] (2018). *What is Java technology and why do I need it*. [online]. Available: https://www.java.com/en/download/faq/whatis_java.xml [Accessed: 18-Oct]

[13] (2018). *C programming Language*. [online]. Available: https://www.techopedia.com/definition/24068/c-programming-language-c [Accessed: 18-Oct]

[14] (2018). *BOA Constructor.* [online]. Available: http://boa-constructor.sourceforge.net/ [Accessed: 18-Oct]

[15] (2018). *Eclipse.* [online]. Available: https://www.eclipse.org/ide/ [Accessed: 18-Oct]

[16] (2018). *PyDev.* [online]. Available: http://www.pydev.org/ [Accessed: 18-Oct]

[17] (2018). *NetBeans IDE Features.* [online]. Available: https://netbeans.org/features/ [Accessed: 18-Oct]

[18] (2018). *What is the Apache HTTP Server Project?*. [online]. Available: http://httpd.apache.org/ABOUT_APACHE.html . [Accessed: 18-Oct]

[19] (2018). *Apache Tomcat*. [online]. Available: http://tomcat.apache.org/ [Accessed: 18-Oct]

[20] (2018). *Oracle GlassFish Server*. [online]. Available: http://www.oracle.com/technetwork/middleware/glassfish/overview/index.html [Accessed: 18-Oct]

[21] E. G. Maida and J. Pacienzia. (2015, diciembre). *Metodologías de desarrollo de software*, [online]. Available FTP: bibliotecadigital.uca.edu.ar Directory: repositorio/tesis File: metodologias-desarrollo-software.pdf

[22] S. A. Asri et al., "Web Based Information System for Job Training Activities Using Personal Extreme Programming (PXP)," in *Journal of Physics Conference Series*, pp. 1-8, Enero 2018.

[23] E. G. Maida and J. Pacienzia. (2015, diciembre). *Metodologías de desarrollo de software*, [online]. Available FTP: bibliotecadigital.uca.edu.ar Directory: repositorio/tesis File: metodologias-desarrollo-software.pdf

[24] V. Y. Rosales-Morales et al., "An analysis of tools for automatic software development and automatic code generation," *Facultad de Ingenieria*, No. 77, pp. 75-87, December 2015.

[25] C. A. Cortes-Camarillo et al., "Análisis comparativo de patrones de diseño de interfaz de usuario para el desarrollo," *Research in Computing Science*, vol 126, pp. 31-41, 2016.

[26] C. A. Cortes-Camarillo et al., “EduGene: A UIDP-Based Educational App Generator for Multiple Devices and Platforms,” *International Journal of Human-Computer Interaction*, pp. 1-23, April 2018.

[27] C. A. Cortes-Camarillo et al., “Atila: A UIDPs-based educational application generator for mobile devices,” *International Conferece on Electronics, Commnications and Computers (CONIELECOMP)*, pp. 1-7, April 2017.

[28] L. N. Sánchez-Morales et al., “Generation of User Interfaces for Mobile Applications Using Neuronal Networks,” *New Perspectives on Applied Industrial Tools and Techniques*, pp. 211-231, June 2017.

[29] S. L. da-Costa et al., “A User Interface Stereotype to build Web Portals,” *9th Latin American Web Congress*, pp. 10-18, 2014.

[30] L. N. Sánchez-Morales et al., “Módulo de generación de aplicaciones multidispositivo a partir del procesamiento de imágenes,” *Research in Computing Science on Computer science and computer engineering*, Issue 9, pp. 81-94, 2015.

[31] A. Halbe and Dr. A. R. Joshi, “A Novel Approach to HTML Page Creation Using Neural Network,” *Procedia Computer Science*, vol 45, pp. 197-204, 2015.

[32] Y. Baveye et al., “Deep Learning for Image Memorability Prediction: The Emotional Bias,” *Proceedings of the 2016 ACM on Multimedia Conference*, pp. 491-495, October 2015.

[33] S. Bianco et al., “Deep learning for logo recognition,” *Neurocomputing*, vol 245, pp. 23-30, 2017.

[34] Y. Liu et al., “Deep learning for pixel-level image fusion: Recent Advances and future prospects,” *Information Fusion*, vol 42, pp. 158-173, 2018.

[35] S. Pang et al., “Deep learning to frame objects for visual target tracking,” *Engineering Applications of Artificial Intelligence*, vol 65, pp. 406-420, 2017.

[36] A. Krishevsky et al., “ImageNet classification with deep convolutional neural networks,” *NIPS’12 Proceedings of the 25th International Conference on Neural Information Processing Systems*, vol 1, pp. 1-9, December 2012.

[37] C. Huang et al., “Large-scale Semantic Web Image Retrieval Using Bimodal Deep Learning Techniques,” *Information Sciences*, pp. 1-39, November 2017.

[38] A. Nedzved et al., “Software Development Technology with Automatic Configuration to Classes of Image Processing Problems,” *Software and Hardware for pattern recognition and image analysis*, vol 23, No 2, pp 269-277, 2013.