ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Diseño e implementación de una red neuronal convolucional para reconocimiento de productos de una empresa de retail

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Telemática

Presentado por:

Emmanuel Fernando Morán Barreiro

GUAYAQUIL - ECUADOR

Año: 2018

DEDICATORIA

El presente proyecto lo dedico a mis padres Ing. Medardo Morán Alvarado y Sra. Carmen Barreiro Ramírez, y de manera especial a mi tío abuelo Dr. Rodolfo Alvarado Mora (+), y familia en general, sin su apoyo no hubiera sido posible llegar hasta este punto de mi vida académica. A mis profesores y compañeros de aula, con quienes aprendí sobre muchos temas durante mis 5 años de pregrado. A mis amigos, los cuales me brindaron el apoyo desde un inicio y sobre todo a las personas que nunca creyeron en mí, por darme una motivación extra para cumplir con este paso y seguir adelante.

AGRADECIMIENTOS

Agradezco a Dios, padre eterno todopoderoso, que en su infinita bondad me brinda muchas oportunidades; también a la empresa privada y al departamento que propuso este proyecto, por hacerme parte en su desarrollo confiando en mis conocimientos y habilidades.

DECLARACIÓN EXPRESA

Los derechos de titularidad y explotación, me corresponde conforme al reglamento de propiedad intelectual de la institución; doy mi consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual.

_____

Emmanuel Morán Barreiro

EVALUADORES

Gabriel Astudillo, **Ph. D.**	**José Córdova, Ph. D.**
PROFESOR DE LA MATERIA	PROFESOR TUTOR

RESUMEN

Reposición y etiquetado de productos son actividades dentro de los supermercados que presentan un problema constante. Los empleados deben estar pendientes de las estanterías o góndolas para reponer los productos, y al inicio de cada día deben verificar que todos los productos estén correctamente etiquetados. Sabiendo que los empleados pueden cometer errores se decide implementar una solución tecnológica a estas actividades que son consideradas sistemáticas. Se implementará un prototipo de modelo de red neuronal artificial convolucional para realizar la identificación de productos en las góndolas, además reconocer en qué estado se encuentran según su presentación, pudiendo ser completo e incompleto. Para desarrollar esto se usó como lenguaje de programación Python junto a sus módulos TensorFlow y Keras que son ampliamente usados en el mundo de aprendizaje de máquina y se construyó algoritmos para revisar la veracidad del modelo de red neuronal sobre los datos adquiridos y parámetros definidos. Se pudo concluir que esta solución es eficiente en la tarea descrita. En entrenamiento tuvo un 100% de precisión para reconocer el producto y 96% de precisión para reconocer el estado del producto. En pruebas con datos no entrenados, infiere correctamente un porcentaje de 100%

Palabras Clave: Retail, Aprendizaje de Máquina, red neuronal convolucional, experiencia de cliente.

ABSTRACT

Replenishment and labeling of products are activities within supermarkets that present a constant problem. Employees must be aware of the shelves to replenish the products, and at the beginning of each day should verify that all products are properly labeled. Knowing that employees can make mistakes, it is decided to implement a technological solution for these activities that are considered systematic. A prototype of a convolutional artificial neural network model will be implemented to carry out the identification of products in the shelves, as well as to recognize in what state they are according to their presentation, being able to be complete and incomplete. To develop this, it was used Python as programming language along with its TensorFlow and Keras modules that are widely used in the machine learning world and algorithms were constructed to check the veracity of the neural network model on the acquired data and defined parameters. It was concluded that this solution is efficient in the task described. In training he had 100% accuracy to recognize the product and 96% accuracy to recognize the status of the product. In tests with untrained data, correctly infers a percentage of 100%.

***Keywords**: Retail, machine learning, convolutional neuronal network, customer experience.*

ÍNDICE GENERAL

ABREVIATURAS

IA	Inteligencia Artificial.
AM	Aprendizaje de Máquina.
AP	Aprendizaje Profundo
API	Aplicación de interface de programación por sus siglas en inglés, Application Programming Interface.
CPU	Unidad de Procesamiento Central por sus siglas en inglés, Central Processing Unit.
GPU	Unidad de Procesamiento Gráfica por sus siglas en inglés, Graphics Processing Unit.
PI	Procesamiento de Imágenes por sus siglas en inglés, Image Processing.
RNA	Red Neuronal Artificial.
RNC	Red Neuronal Convolucional.
SBC	Computadora de placa reducida por sus siglas en inglés, Single-Board Computer.
SDK	Herramientas para desarrollo de software por sus siglas en inglés, Software Development Kit.
TPU	Unidad de Procesamiento de Tensores por sus siglas en inglés, Tensor Processing Unit.
VC	Visión Computacional.

SIMBOLOGÍA

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

El tipo de ´´*Retail*´´ denominado **supermercado**, se define según el diccionario de la lengua española como: ”establecimiento comercial de venta al por menor en el que se expenden todo género de artículos alimenticios, bebidas, productos de limpieza, etc., y en el que el cliente se sirve a sí mismo y paga a la salida; con esta clara definición, se entiende al cliente como lo más importante que se tiene” [1] ; y por eso, brindarle comodidades y mejoras a ellos es una de las prioridades en este tipo de empresa.

Una empresa retail, que de ahora en adelante se la denominará ´´*El Supermercado*´´, es de tipo supermercado y de conocimiento nacional con clientes objetivo de un estrato económico medio-bajo. Se enfoca directamente en las ventas al por menor y en grandes cantidades. Además, El Supermercado acostumbra a tener variadas ofertas en sus secciones con el fin de llamar la atención de sus clientes y dar una alta rotación a la mayor cantidad de sus productos. Una observación de El Supermercado es que gran parte de sus ventas son por impulso, lo que significa que el cliente compró en la tienda o aumento algo a su carrito por haber observado el producto indicado, en el lugar indicado y en el momento indicado.

El Supermercado, con el propósito de darle una mejor atención a sus clientes y de la misma forma generar información valiosa para análisis y toma de decisiones, desea implementar una solución tecnológica a las problemáticas de *”reposición de productos”* y *”actualización y mantención de etiquetado de los productos”* en los lineales de góndolas [1] de sus locales.

En este proyecto se utilizará redes neuronales convolucionales para brindar solución a una parte importante del prototipo de bajo costo que desea implementar El Supermercado para la problemática descrita en los siguientes literales.

[1] Los lineales de góndola son cada horizontal de la góndola donde se colocan a exhibir los productos. Normalmente una góndola puede tener de entre 4 a 8 lineales de diversas formas para colocar diferentes tipos de productos.

1.1 Descripción del problema

Varios jefes de los locales informan la existencia de quejas por parte de los clientes en dos puntos:

1. Los productos no son repuestos de manera eficiente, ocasionando que existan espacios vacíos en los lineales de las góndolas.
2. Varios clientes tienen el desagrado de tomar un producto y en la caja descubren que el precio el cual se mostraba en la góndola no es el correcto.

Estos problemas se presentan en gran porcentaje de los locales a nivel nacional, y a su vez en varias empresas retail de tipo supermercado; y son dados por las actividades de *"reposición de productos"* y *"actualización y mantención de etiquetado"*. Ambas actividades son realizadas de manera manual por el personal asignado a realizarlo. A continuación, se detallan las actividades antes mencionadas.

- La actividad de etiquetado es realizada en horario fuera de atención al cliente, normalmente bien temprano en la mañana antes de iniciar la jornada, porque los envíos de información (cambios de precios, stocks, ventas, etc.) son proporcionados al cierre de la tienda del día anterior, alrededor de media noche. El personal asignado deberá verificar que las etiquetas de los precios coincidan con los productos que están siendo presentados, y también sacar y poner etiquetas nuevas de ser necesario, tanto por desgaste de etiqueta o por ofertas.
- La actividad de reposición es realizada durante la jornada de ventas, pues si un producto se queda con pocas existencias en el lineal debe ser repuesto rápidamente para que los clientes puedan comprarlo. Al momento de existir ofertas de un producto, los locales se preparan para la demanda que va a existir por parte de los clientes asignando más espacio en los lineales a los productos que presentan la oferta y así evitar tener que reponer los productos muy seguidamente. Pero lo descrito es para una oferta, para el caso de productos sin oferta, el personal debe estar atento para, de ser requerirse, reponer el producto oportunamente.

Estas actividades son críticas pues están ligadas a que el cliente pueda hacer la compra del producto, pero por ser realizadas manualmente darán fallas continuamente, aumentando el número de clientes que son afectados en no encontrar los productos que buscan o escoger productos con precio diferente al mostrado.

1.2 Justificación del problema

Al tener una base de ventas de alta rotación, brindarles a sus clientes existencias de productos en todo momento se vuelve crítico para El Supermercado. Su principal misión es vender, y sin existencias en los lineales de las góndolas esta misión no se puede cumplir. La reposición es una actividad crítica, pero tiene un complemento para lograr que las ventas sean exitosas, y es mantener un etiquetado correcto y actualizado en toda jornada de ventas.

Como se indicó anteriormente, El Supermercado es un retail de autoservicio, y los clientes escogen los productos que van a comprar; por esta razón es imperativo que el cliente pueda observar los productos en buen estado (perchados correctamente), y con su etiqueta correcta presentando el precio al cual lo comprarán.

Realizando estas actividades con menor o ningún porcentaje de error, se evitará que el cliente se encuentre con situaciones poco favorables para la venta. Una de estas situaciones es cuando no existen productos por falta de reposición, en esta situación el cliente optará por salir del local en búsqueda del producto en otro lugar lo cual se distingue como una pérdida de venta. Otra situación puede ser cuando se le factura un producto por un precio más elevado del que se le presentó en la góndola, en esta situación el cliente podría desconcertarse, enojarse y hasta juzgar al local, ocasionando que el cliente no deseé volver a comprar en el mismo local. Además, actualmente también se pueden presentar desconformidades de estos casos en las redes sociales, lo cual genera prejuicios por parte de otros compradores potenciales.

1.3 Alcance

Para la entrega de este proyecto integrador se mide como alcance la elaboración de un modelo de red neuronal convolucional para reconocimiento de productos y estado de sus existencias en el frente de lineal que ocupa en la góndola (los estados posibles son *completo* e *incompleto*). Esta red neuronal será verificada con un máximo de 10 productos para evitar la prolongación de la toma de datos. La presentación se realizará en comando de líneas con un menú que facilite probar su veracidad de reconocimiento.

Dentro del reporte que emitirá la red al ser probada, deberá brindar los siguientes mensajes:

- Si no existe un producto, deberá emitir el mensaje de ”PERCHAR”, indicando que existe un vacío en el lineal.
- Si existe un producto y su estado determinado es ”INCOMPLETO”, deberá emitir el mensaje de ”REVISAR ESTADO EN GÓNDOLA”, indicando que hay existencias, pero su estado no es el óptimo para presentar al cliente.
- Si existe un producto y su estado determinado es ”COMPLETO”, deberá emitir el mensaje de ”PRODUCTO CORRECTO”, indicando que no es necesario revisar tal producto.

1.4 Objetivos

1.4.1 Objetivo General

Diseñar e implementar una red neuronal convolucional para reconocimiento de productos en un lineal de góndola de locales de empresa retail.

1.4.2 Objetivos Específicos

1. Implementar un algoritmo para recolección de datos personalizados al espacio del problema para entrenamiento de la red neuronal.
2. Desarrollar un prototipo físico de un lineal de góndola para extracción de datos de entrenamiento, validación y prueba de la red neuronal.
3. Seleccionar una arquitectura de red neuronal convolucional y evaluar los hiperparámetros adecuados para reconocer los productos en su espacio.
4. Evaluar entrenamiento realizado a la red neuronal convolucional.
5. Desarrollar una herramienta de visualización para mostrar resultados de la predicción de la red neuronal convolucional y mensajes de acción para el personal de El Supermercado.

CAPÍTULO 2

2. MARCO TEÓRICO

2.1 Inteligencia Artificial

La Inteligencia Artificial (IA) es un campo de estudio que involucra a varias tecnologías que actualmente son objeto de desarrollo e investigación. Una definición concisa sería: *esfuerzo por automatizar actividades intelectuales normalmente ejecutadas por humanos* [2].

En la actualidad, se apunta a la creación de soluciones con el uso de inteligencia artificial, proponiéndose automatizar procesos para mejorar la productividad y la experiencia del cliente [3].

2.2 Aprendizaje de Máquina

El aprendizaje de máquina (AM) es una parte del estudio de la Inteligencia artificial, que se fundamenta en el concepto de *computadora de propósito general*; ideado por Lady Lovelace y luego complementado por Alan M. Turing quien desarrolla una investigación a partir de sus ideas [4]. Este paradigma de programación tiene en su fondo una fuerte base matemática y estadística ya que utiliza algoritmos técnicos para minimizar el error en la optimización que se desea realizar [5].

Además, AM cambia el paradigma de programación, el cual se enfoca en dar reglas a partir de datos y respuestas, en lugar de entregar respuestas a partir de datos y reglas como se puede ver en la Figura 2.1. Así mismo se puede apoyar del estudio de visión computacional (VC) y el procesamiento de imágenes (PI) para dar como resultado una fuerte herramienta para resolución de problemas perceptuales de reconocimiento y clasificación.

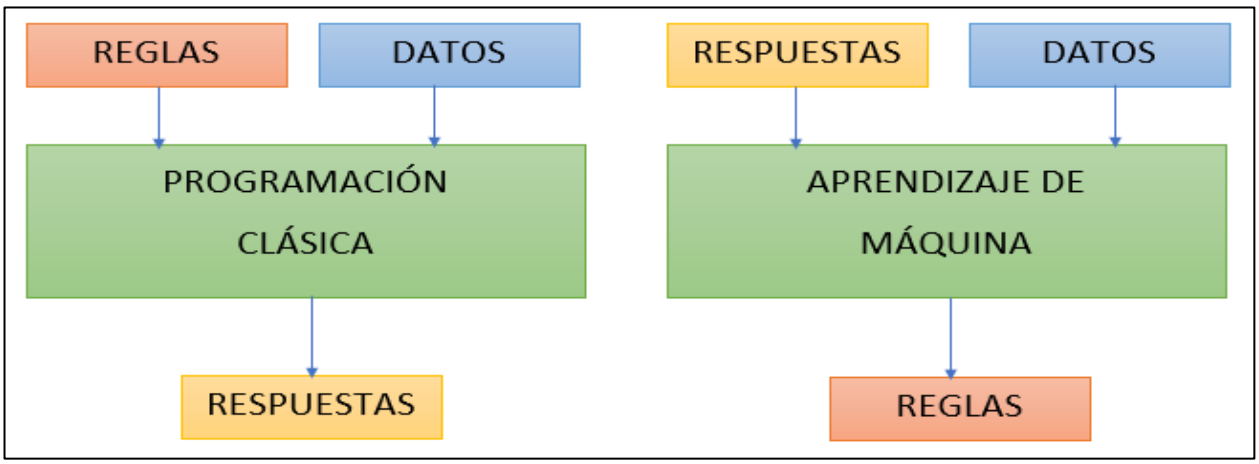


Figura 2.1: Programación clásica vs aprendizaje de máquina.

2.3 Neurona Artificial

Una neurona artificial es un concepto matemático desarrollado para llevar a cabo tareas sencillas presentado por Warren McCulloch y Walter Pitts [6].

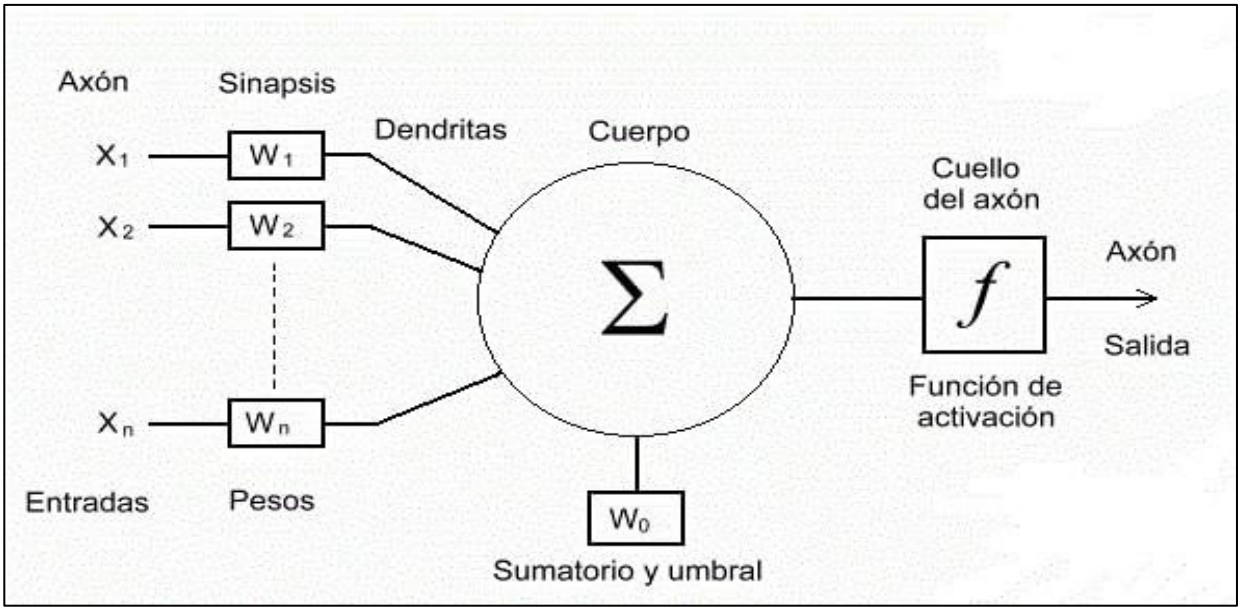


Figura 2.2: Representación del modelo de neurona artificial analizado por W. McCulloch y W. Pitts en 1943.

En la Figura 2.2 se puede apreciar el concepto matemático de la neurona artificial. La salida de esta neurona se determina por la función:

$$Y = f\left(\sum_{i=1}^{n} w_i x_i\right)$$

La función de activación puede ser elegida dependiendo del problema a resolver.

2.4 Redes Neuronales Artificiales

Las redes neuronales artificiales (RNA) son algoritmos desarrollados para imitar el trabajo que realizan las redes neuronales del cerebro humano [7]. Actualmente, gracias a la disponibilidad de dispositivos capaces de soportar la implementación de estos algoritmos, se consideran técnicas de estado-de-arte para muchas aplicaciones de aprendizaje de máquina.

Las RNA se han implementado en varios lenguajes de programación; su uso es recomendado para el rápido y fácil desarrollo de soluciones con AM [8]. Se diseñan de manera similar a la que se presenta en la Figura 2.3. Esta representación es también llamada **perceptrón multicapa**, en la cual se aprecian tres tipos de capas (de entrada, oculta y de salida), cada una de estas tiene cierto número de *neuronas* definidas según la necesidad del problema.

Las neuronas, en la capa de entrada y de salida tienen el trabajo respectivo de ingresar el conjunto de datos pertinente a la red y arrojar un resultado (único o múltiple). Las neuronas en las múltiples capas ocultas de la red realizan conexiones entre capas (capa ”n” se conecta con capa ”n-1” y capa ”n+1”), con la finalidad de delimitar un camino según la información ingresada. De acuerdo al número de capas ocultas se puede indicar, que tan profunda es la red y dependiendo de esto se dan varias aplicaciones.

Existe una parte del aprendizaje de máquina donde las redes neuronales creadas son extensas y se denominan redes neuronales profundas.

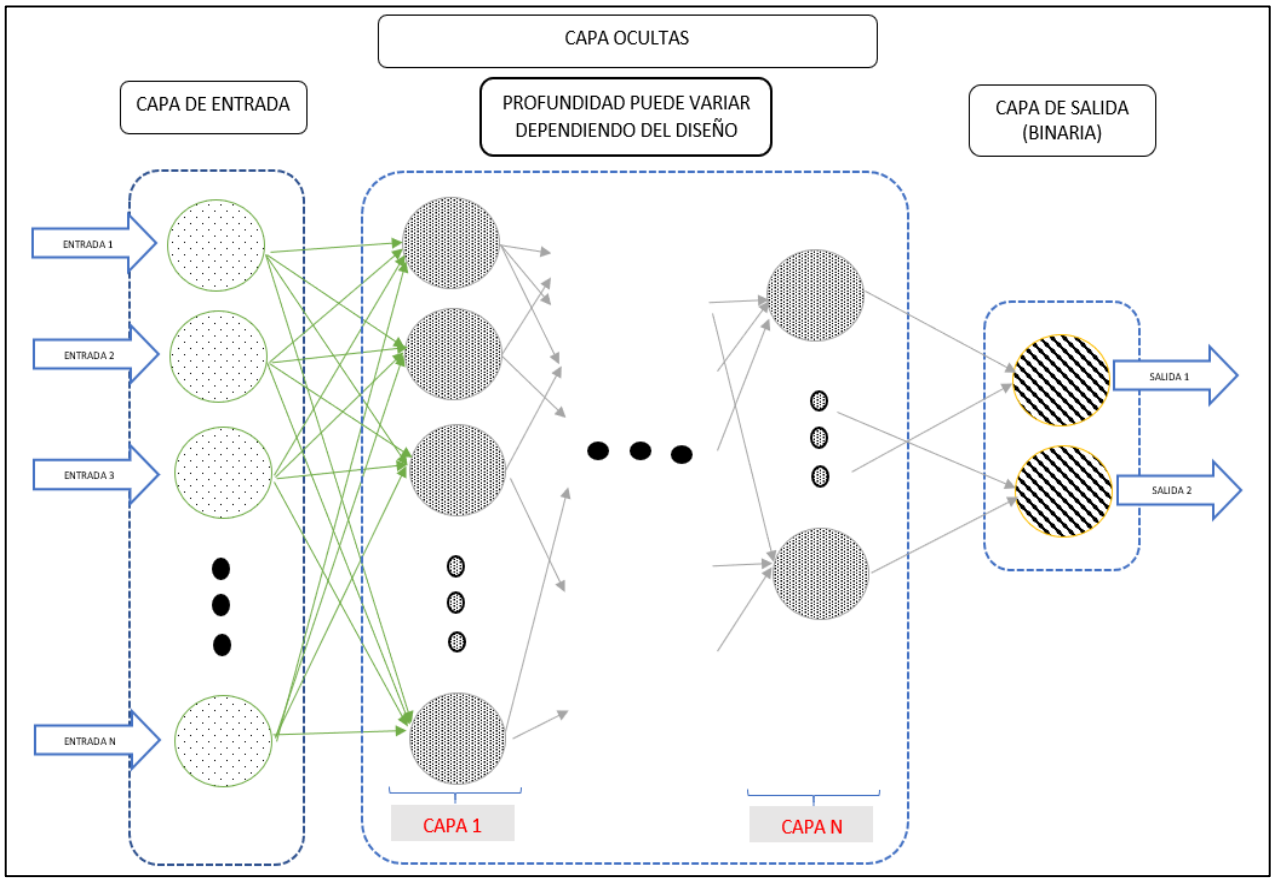


Figura 2.3: Representación básica del diseño de una red neuronal artificial, perceptrón multicapa.

2.5 Redes Neuronales Artificiales Convoluciones

Una red neuronal artificial convolucional, o solamente red neuronal convolucional (RNC), es un tipo de implementación de red neuronal que tiene como objetivo aprender características de orden alto de la información por medio de convoluciones. Es ampliamente utilizada para solucionar problemas de visión [9] [10] (clasificación de imágenes, reconocimiento de objetos, etc.), y también para análisis de palabras y análisis sentimental [11] [12].

Este tipo de red fue inspirada por la corteza visual de los animales [10]. Su diseño de implementación puede apreciarse en la Figura 2.4, donde se observan cambios de dimensiones en los datos ingresados. Este cambio es parte de la implementación de la

neurona convolucional y depende de los filtros que se quiera implementar durante el modelamiento de la red neuronal.

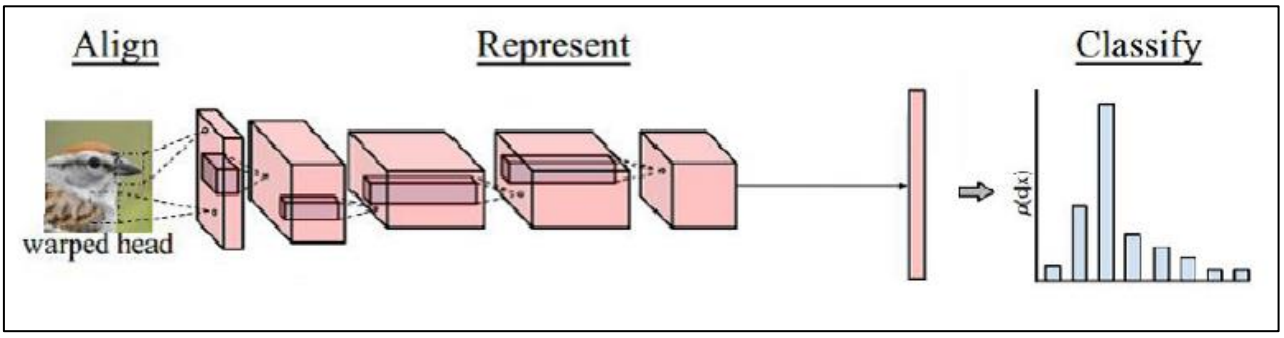


Figura 2.4: Proceso de clasificación realizado en una red neuronal convolucional artificial.

Como su nombre lo indica, este tipo de redes neuronales usan la operación de convolución.

2.5.1 La convolución

es un operador matemático que transforma dos funciones **f** y **g** en una tercera función que en cierto sentido representa la magnitud en la que se superponen f y una versión trasladada e invertida de g.

Fórmula en caso discreto:

$$y(t) = x(t) * h(t) = \sum_{\tau \to -\infty}^{\infty} x(\tau) h(t - \tau)$$

Fórmula en caso continuo:

$$y(t) = x(t) * h(t) = \int_{\tau \to -\infty}^{\infty} x(\tau) h(t - \tau)$$

2.5.2 Propiedades de las redes neuronales convolucionales

Las propiedades clave de las redes neuronales son las siguientes:

1. Invariación de traslado de patrones aprendidos
2. Jerarquía espacial de patrones

La característica de invariación de traslado de patrones aprendidos es una de las principales razones por las que las redes convolucionales son líderes en competencias mundiales de aprendizaje de máquina para clasificación de imágenes [13]. Con esta característica los patrones encontrados durante el entrenamiento serán reconocidos en cualquier parte de los nuevos datos. A modo de ejemplo, si la RNC reconoce un caballo en una esquina superior derecha, lo reconocerá nuevamente en otra imagen en la esquina inferior izquierda. La característica de jerarquía espacial de patrones permite reconocer patrones complejos a medida que se va profundizando en la red mediante la combinación de patrones más simples en las primeras capas, ver Figura 2.5

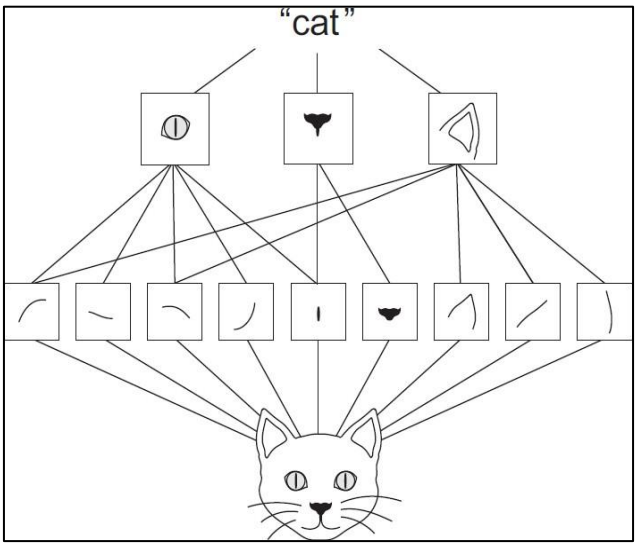


Figura 2.5: Patrones simples de las primeras capas se combinan para formar patrones más completos. [24]

2.6 Programación de redes neuronales artificiales

En la actualidad existen muchos lenguajes como Java, C/C++, R, Lisp y Python, que ofrecen soluciones abstractas para la implementación de redes neuronales de manera rápida. En este proyecto se hará uso del lenguaje Python.

2.6.1 Python

Python es un lenguaje de programación de alto nivel que permite el trabajo rápido con sintaxis de código sencilla y simple. En la actualidad existen dos versiones; la versión 2.7.x que es ampliamente usada en las distribuciones de Linux de código abierto y la versión 3.x con su última actualización a 3.6.5 hasta la fecha. Python es un lenguaje de propósito general y uno de los más usados en todo el mundo debido a que abarca un gran número de posibles campos de aplicación [14] [15].

2.6.1.1 TensorFlow

Es una biblioteca de software de código abierto para el cálculo numérico de alto rendimiento; su arquitectura flexible permite una fácil implementación de computación en una variedad de plataformas (CPU, GPU, TPU) y desde escritorios hasta clústeres de servidores y dispositivos móviles y periféricos. Desarrollado originalmente por investigadores e ingenieros del equipo Google Brain dentro de la organización AI de Google, cuenta con un sólido respaldo para el aprendizaje de máquina y el aprendizaje profundo y el núcleo de computación numérica flexible se utiliza en muchos otros dominios científicos [16].

2.6.1.2 Keras

Es una API de redes neuronales de alto nivel, escrita en Python y capaz de ejecutarse sobre TensorFlow, CNTK o Theano. Fue desarrollado con un enfoque para permitir la experimentación rápida y

poder pasar de la idea al resultado en el menor tiempo posible para hacer una buena investigación [17].

Las facilidades de Keras son:

- Permitir la creación fácil y rápida de prototipos (a través de la facilidad de uso, modularidad y extensibilidad).
- Admitir redes convolucionales y redes recurrentes, así como sus combinaciones.
- Ejecutar sin problemas en CPU y GPU.

3. METODOLOGÍA

3.1 Definición del producto

El producto por entregar de este proyecto es el modelo entrenado de una red neuronal convolucional para reconocimiento de productos con la facilidad de ser preciso, escalable y re-entrenable.

3.2 Esquema de Trabajo

La elaboración del proyecto tiene 5 etapas de desarrollo. Estas etapas están presentadas en la Figura 3.1.

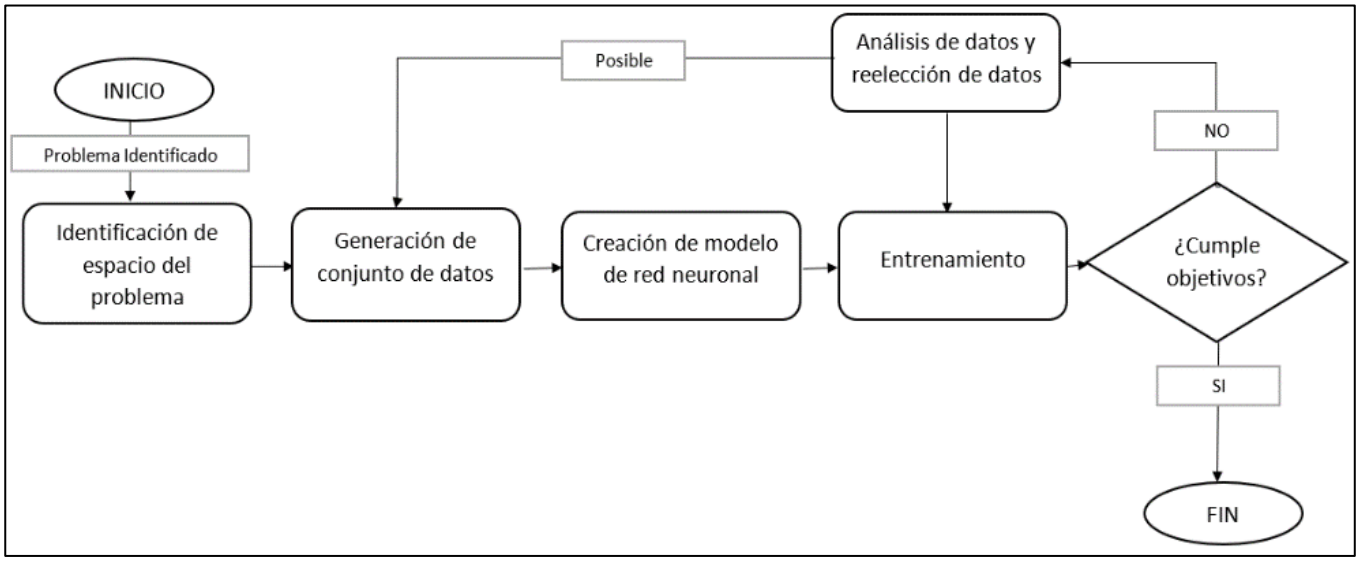


Figura 3.1: Esquema de trabajo.

3.2.1 Etapa 1: Identificación de espacio del problema

Una vez establecidos los parámetros del problema se visitó al local donde se llevarán a cabo las pruebas a futuro. En esta etapa se reconocieron las diferentes características de las góndolas y productos para ser tomadas a consideración.

Durante esta etapa se identificó que los productos pueden variar su posición, por lo tanto, son independientes de su locación y de otros productos aledaños. En consecuencia, se decidió definir un límite entre producto y producto, siendo este las etiquetas de los productos. En la Figura 3.2.a se puede notar la manera definida para etiquetar los productos, mientras que en la Figura 3.2.b se identifican dos maneras erróneas de etiquetar un producto; una es poniendo la etiqueta bajo el producto, pero en la mitad del espacio del frente de lineal de góndola que ocupa el producto; y la otra es no poner la etiqueta del producto.

a) etiquetas colocadas de manera correcta. b) etiquetas colocadas de manera incorrecta.

Figura 3.2: Productos perchados en un lineal de góndola real.

Utilizando la forma de etiquetar los productos vista en la Figura 3.2.a, permite realizar un recorte fácil de la sección que es de interés antes de enviar la imagen por la red neuronal.

3.2.2 Etapa 2: Generación de conjunto de datos

Con los requerimientos descritos anteriormente, se inició la adquisición de muestras para el diseño y modelamiento de la red neuronal, las cuales deben ser representativas con respecto a la población objetivo (productos situados en los lineales de las góndolas).

En la Figura 3.3 se aprecia la maqueta desarrollada para el proyecto que se utilizó para la adquisición de datos. La maqueta auxiliar presenta en los bordes superior e inferior canaletas de aluminio (a) que se conectan a una ”C” de aluminio (b) con ayuda de ruedas similares a las usadas en puertas corredizas (c); sobre la ”C” de aluminio se montó una cámara (d) para ejecutar la toma de datos (fotos). Este sistema se diseñó para simular el recorrido ejecutado por una persona mientras está visitando la tienda y revisando las góndolas.

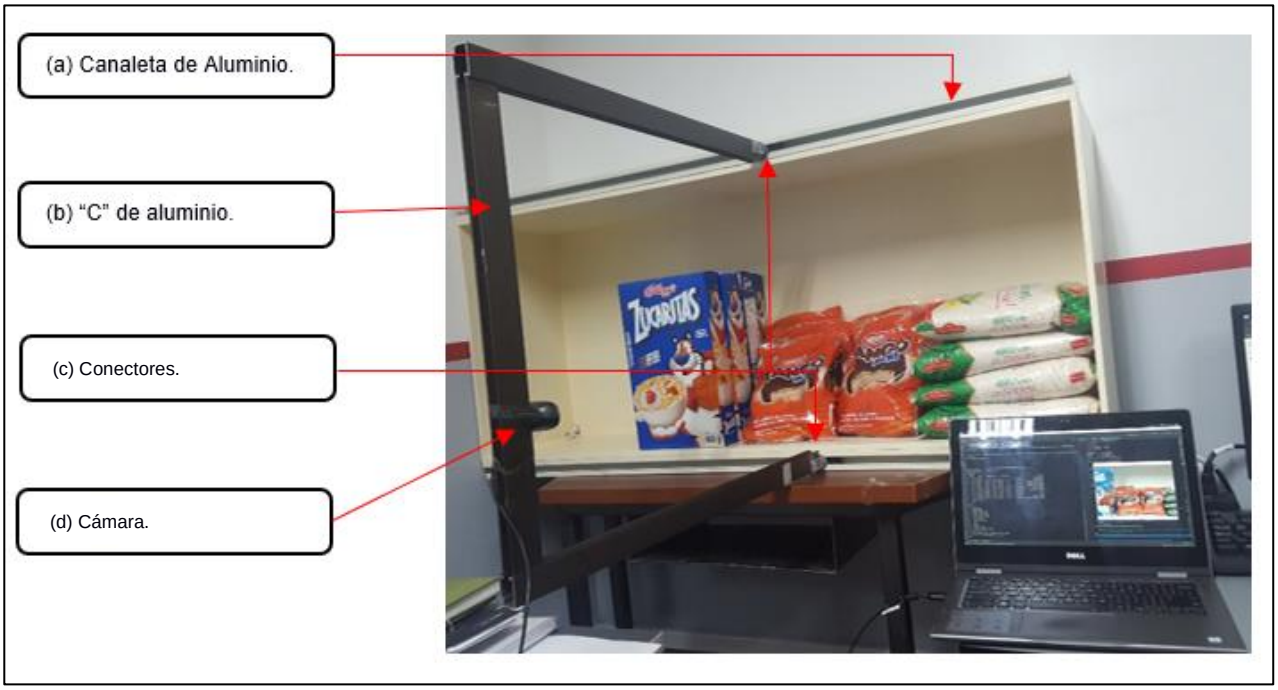


Figura 3.3: Maqueta auxiliar para adquisición de datos.

Los datos para este proyecto son fotos a color de productos en la maqueta auxiliar de tal manera que se asemejen a los productos existentes en los lineales de góndola.

En la Figura 3.4.a se aprecian los datos crudos (foto de toma completa) que contienen información considerada ruido, como:

- Espacio vacío en la parte superior.
- Parte del conector de la ”C” de aluminio y las canaletas de aluminio en la parte inferior de la maqueta.
- Gran parte de otros productos a ambos lados.

Figura 3.4: Secuencia de preprocesamiento de los datos adquiridos.

Para este proyecto se trabajó con los datos (fotos) procesados en memoria como matrices multidimensionales. Por tanto, los datos debieron ser cortados (Figura 3.4.b) y redimensionados a una medida de anchura y altura específica (Figura 3.4.c) para poder empatar con la entrada de la RNC.

Teniendo en cuenta que se presentarán casos donde la diferencia de patrones será en los niveles de coloración (escala RGB, modelo de colores a base de coloración rojo, verde y azul) se trabajó con los datos en colores, dejando la matriz multidimensional de cada dato en memoria con las dimensiones de 299 para anchura, 299 para altura y 3 para canales. ver Figura 3.5.

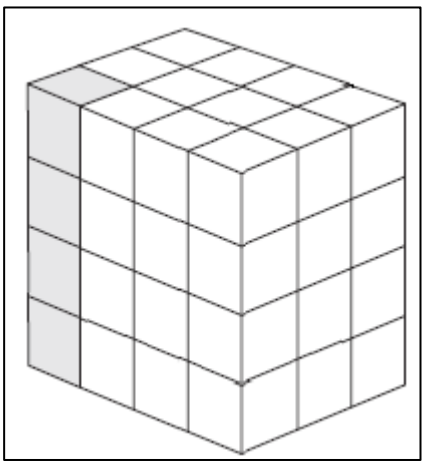


Figura 3.5: Visualización ejemplo de un dato adquirido (foto) en memoria.

Para la ejecución de la adquisición de datos se realizó un algoritmo denominado ”data_generator.py” con estilo de menú de ayuda. Este menú permite escoger de entre todos los productos uno al cuál se procederá a adquirir datos (tomar fotos) y administrar esta acción para no caer en el error de adquirir muchos datos de un solo tipo y sesgar el modelo.

Para cumplir con las actividades de entrenamiento y prueba de la red neuronal se adquirió datos para separar en 3 tipos de conjuntos de datos:

- **Conjunto de entrenamiento**, es el conjunto de datos al cual la red neuronal recorre varias veces en búsqueda de patrones para aprenderlos. Es preciso que este conjunto sea lo más general posible.
- **Conjunto de validación**, es aquel conjunto de datos que se envía junto al de entrenamiento para que, al terminarse cada recorrido completo sobre este, se realicen pruebas y se presenten porcentajes de error y exactitud.
- **Conjunto de prueba**, es aquel conjunto de datos que una vez entrenada la red neuronal, se utiliza para probar si puede clasificar de manera correcta datos que nunca toco durante su entrenamiento. Es decir, con este se confirma la generalización desarrollada por la red neuronal.

3.2.3 Etapa 3: Creación de Modelo de Red Neuronal

Antes de iniciar el código de la red neuronal se estableció correctamente un ambiente de desarrollo. Esto implicó instalar software y dependencias compatibles necesarias para realizar el proyecto. Para esto, se adjunta la Tabla 3.1 que describe los elementos tanto en hardware y software que serán requeridos para la elaboración, entrenamiento y prueba de la red neuronal.

Una vez establecido el ambiente de desarrollo, se dio inicio a la implementación en código de la red neuronal. En este proyecto se creó una clase de apoyo para la construcción de una red neuronal. Una clase es un modelo que define un conjunto de variables y métodos apropiados para operar con dichos datos. Cada objeto creado a partir de una clase se denomina instancia. La clase creada se denominó ”Product_Detector.py”, en ella se definen la inicialización de varios parámetros vitales para entrenar una red neuronal, así como métodos de auxilio para realizar acciones con la red como: guardado, carga, entrenamiento, validación y otros.

Tabla 3.1: Descripción y especificaciones técnicas de elementos del proyecto.

Elemento	Descripción
Logitech c310	Cámara HD 720p, conexión USB 2.0, 5mp
Computadora de alto rendimiento	252gb SSD, GPU NVIDIA GeForce
Ubuntu 16.04	Sistema
Drivers nvidia 384	Drivers compatibles con la tarjeta gráfica usada y los elementos para aceleración de GPU con frameworks usados.
CUDA 8.0	Nvidia Cuda Toolkit, SDK de computo con GPU,en su versión 8.0 GA2.
cuDNN 6.0	Librería para aceleración de GPU
Python 3	Versión de Python 3.5.2 estable.
Tensorflow-gpu 1.4.1	Framework en lenguaje Python para manejo de tensores en redes neuronales en su versión para GPU s 1.4.1
Keras 2.1.4	API de alto nivel para desarrollo de redes neuronales en su versión 2.1.4

Los modelos de redes neuronales se construyen uniendo varias capas de neuronas artificiales, como se lo visualizó en la Figura 2.3.

Para este proyecto se decidió realizar pruebas con los modelos de redes neuronales convolucionales DENSETNET121, DENSENET169, DENSENET201 [18], INCEPTIONv3 [19], INCEPTION-RESNETv2 [20], VGG16, VGG19 [21], XCEPTION [13]. Estas redes son reconocidas por su efectividad en reconocimiento de objetos utilizando los bloques de neuronas convolucionales en diversos diseños.

Los modelos de redes antes mencionados tienen soporte dentro del módulos de aprendizaje profundo de Python, keras, y brindan la facilidad de ágil implementación modificando sus hiperparámetros.

3.2.4 Etapa 4: Entrenamiento

Luego de haber definido una arquitectura idónea para la red neuronal, se realizó el lanzamiento de entrenamiento, que resultó en dos archivos que sirven para cargar el modelo junto a sus pesos aprendidos durante el entrenamiento. Estos archivos permiten que el producto final no se tenga que entrenar cada vez que se inicie, pues no sería lógico entrenar el producto final cada vez que inicia porque esta actividad puede tardar desde minutos hasta horas, y en algunos casos días, aún con computadoras de gran procesamiento.

Con la ayuda de la computadora de alto rendimiento, que tiene sus características detalladas en la Tabla 3.1, se realizaron los entrenamientos. Se seleccionó la variable de radio de regularización que será usada como regularizador de kernel con la función cuadrática, esto se aplica en las capas finales del modelo diseñado. Esta variable es usada para mejorar el éxito en la tarea de reconocimiento de objetos ayudando a disminuir la dispersión de los pesos en el modelo [22].

En la Figura 3.6 se puede observar el número de parámetros a ser entrenados para una red usando el modelo de red neuronal Xception. Se distinguen parámetros entrenables y no entrenables, de igual manera se nota la cantidad de “samples” que son el número del conjunto de datos adquirido; 4325 para el conjunto de entrenamiento y 127 para el conjunto de validación.

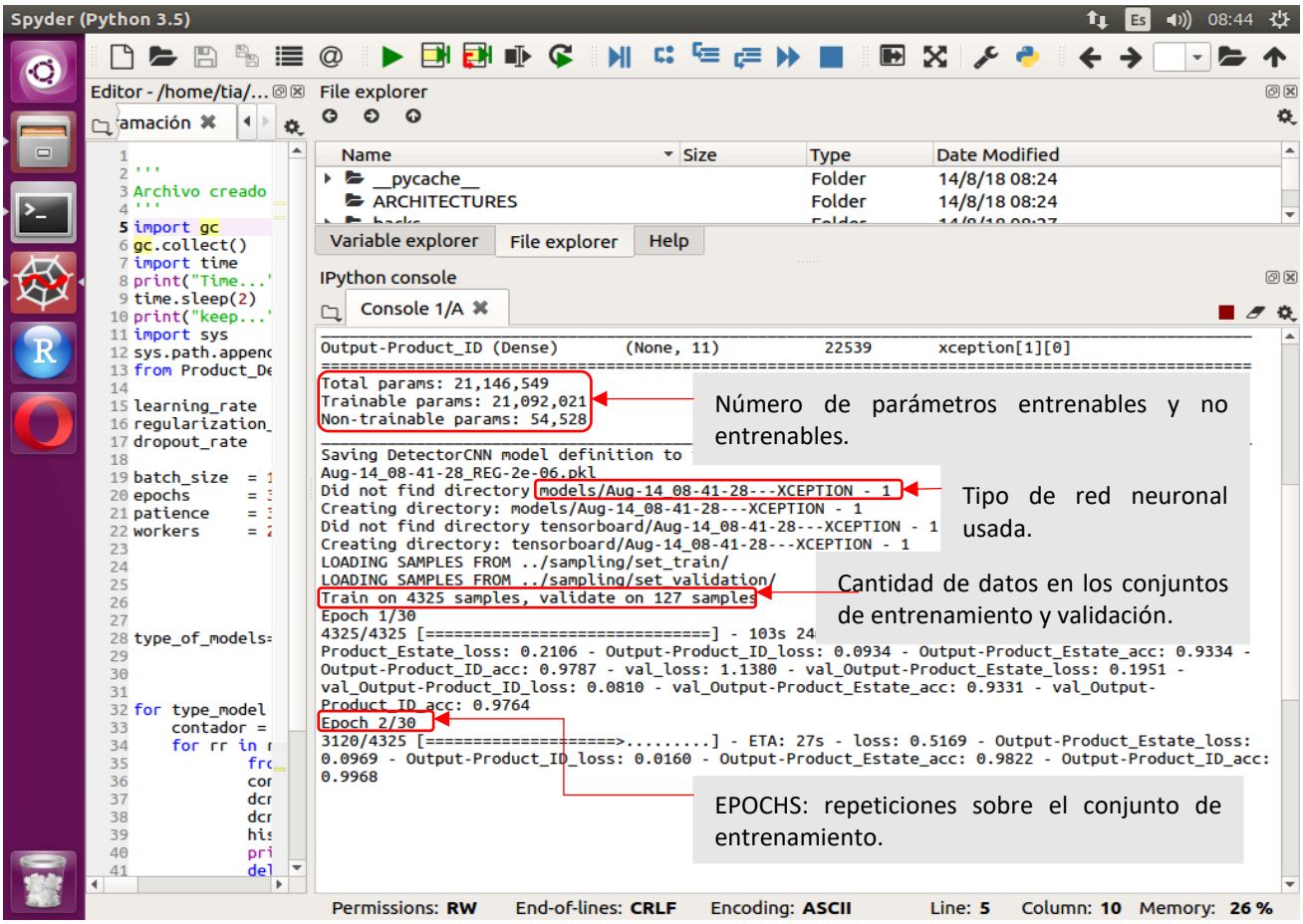


Figura 3.6: Entrenamiento de red neuronal en computadora de alto rendimiento usando software SPYDER3.

3.2.5 Etapa 5: Análisis de datos y reelección de datos

Estos archivos generados del entrenamiento serán usados junto a un algoritmo que desplegará un menú para validar el funcionamiento correcto de la red neuronal entrenada. El flujo de este menú se detalla en la Figura 3.7.

En caso de que la red neuronal no cumpla con el objetivo principal (clasificar de manera correcta los objetos), se regresará a la etapa 4, donde se

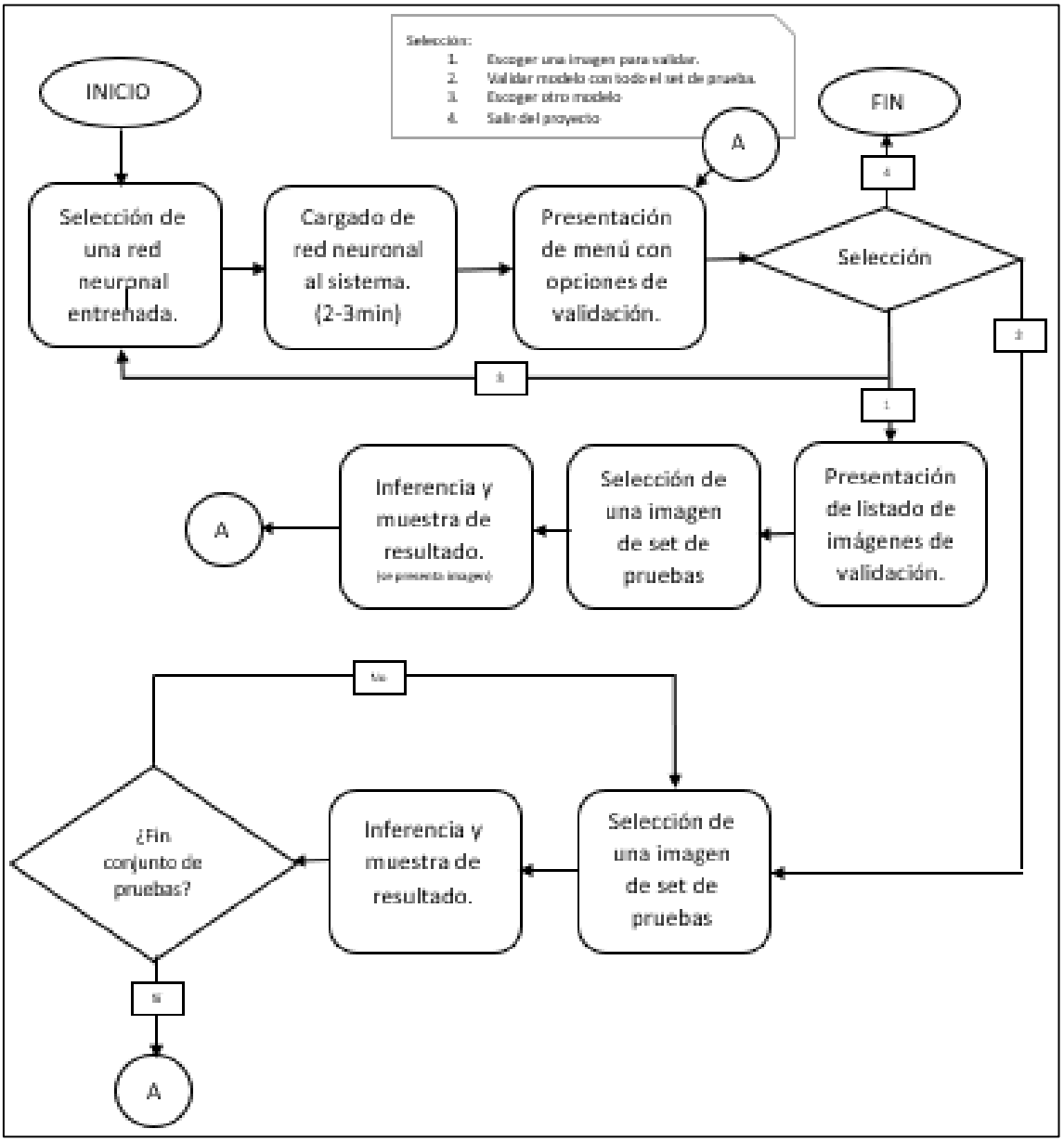


Figura 3.7: Secuencia de algoritmo para validar una red neuronal entrenada.

seleccionarán nuevamente las variables de aprendizaje necesarias y en algunos casos será necesario inclusive la modificación de la arquitectura de la red neuronal. Puede darse el caso que durante este ciclo entre la etapa 4 y 5, se presenten inconformidades en los datos. Para este caso, será necesario volver a la etapa 2, y recolectar más o nuevos datos para alimentar a la red neuronal durante su entrenamiento.

CAPÍTULO 3

4. RESULTADOS Y ANÁLISIS

4.1 Algoritmo para adquisición de datos

Como parte del proyecto se definió la realización de un algoritmo que dé soporte para adquirir los datos necesarios del entrenamiento de la red neuronal. Se presenta entonces el esquema que se eligió para este proceso de adquisición de datos en la Figura 4.1.

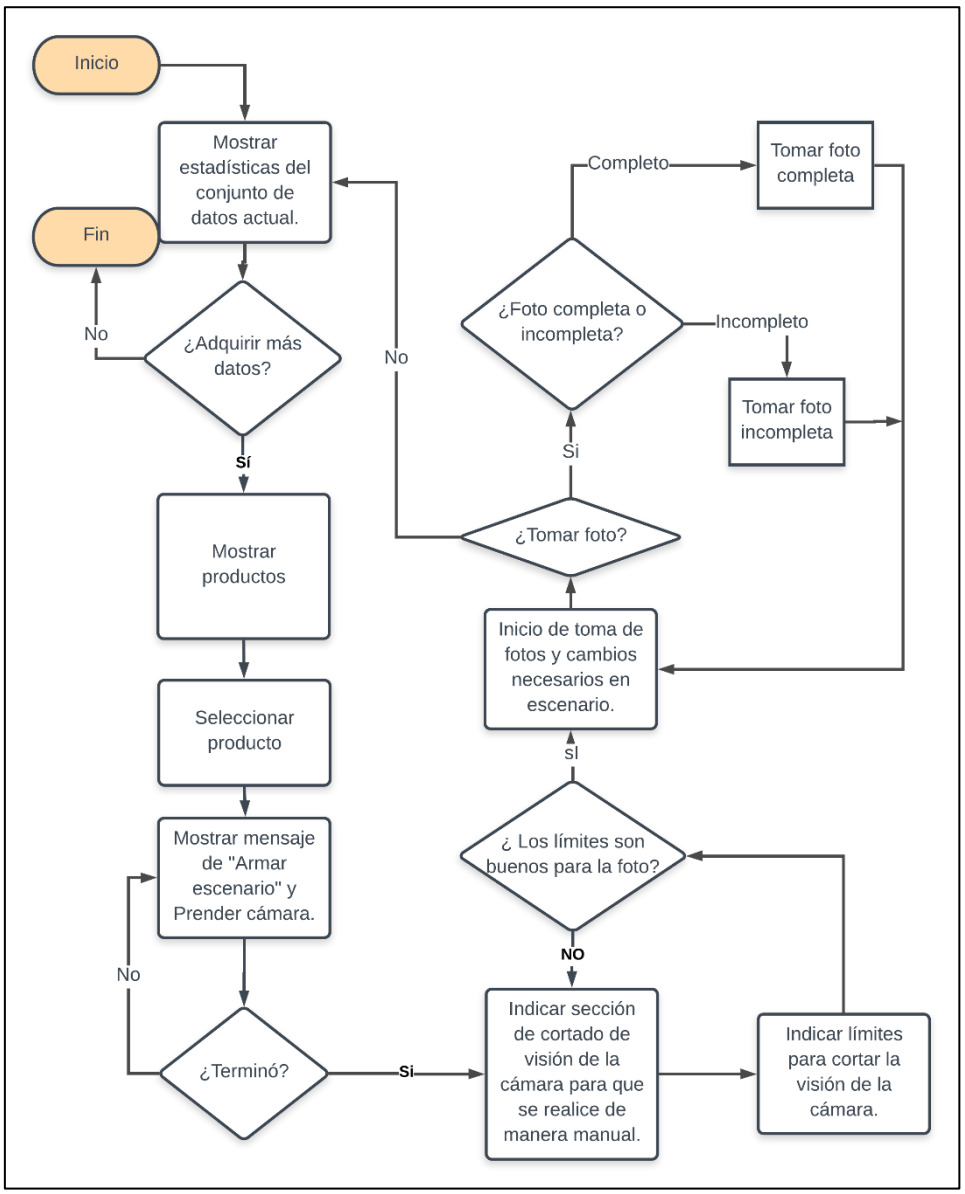


Figura 4.1: Esquema de adquisición de datos.

Cabe recalcar que los datos para este proyecto son fotos de productos, los cuales deberán pasar por un pre-proceso antes de poder servir de alimento al entrenamiento de la red neuronal.

4.1.1 Presentación de datos actuales y selección de producto para muestreo

El algoritmo generado fue previsto para brindar soporte completo de las necesidades de los datos que se van adquiriendo, por esto lo que se presenta al inicio es una serie de estadísticas sobre los datos en el directorio donde son guardados.

En esta fase y con ayuda de la presentación antes mencionada se podrá discernir que producto está quedando rezagado en la adquisición de datos. También presenta cuál de los estados tiene mayor porcentaje entre los datos de un mismo producto.

Esto da paso a definir cantidad de datos necesarios por cada producto, brindando la posibilidad, más adelante, de controlar los datos que se tomen de manera dinámica. En la Figura 4.2 se puede observar cómo se selecciona uno de los productos para iniciar el proceso de la adquisición de datos y se prende la cámara que facilitará establecer un escenario con el producto a mostrar en medio de dos productos aleatorios. Este escenario es construido pensado en cómo son colocados los productos en los lineales de góndola.

Figura 4.2: Proceso de adquisición de datos: paso de colocación de escenario con producto seleccionado.

4.1.2 Pre-proceso estático de las imágenes

En la Figura 4.4 se aprecia la parte del menú de adquisición de datos donde de manera manual se establecen límites superior inferior izquierdo y derecho para la visión de la cámara

4.1.3 Toma de datos de producto en diferentes estados

Una vez definidos los límites de la cámara, se continúa con el algoritmo para capturar las imágenes, redimensionarlas y guardarlas, para finalmente presentar nuevas estadísticas sobre el conjunto de datos adquirido.

Se notará en la Figura 4.3 que el menú de selección para tomar datos lo puede hacer tanto con estado de completo como de incompleto.

```
============== ELEGIR SIZE PARA CROP =============
ver imagen...

            left_row    =  0
            right_row   =  480
            left_col    =  0
            right_col   =  640

Ingresar S para terminar:
:
change left_row? y/n:
:y
Actual left_row: 0 change to...
Ingresar numero: 100
change right_row? y/n:
:y
Actual right_row: 480 change to...
Ingresar numero: 440
change left_col? y/n:
:y
Actual left_col: 0 change to...
Ingresar numero: 120
change right_col? y/n:
:y
Actual right_col: 640 change to...
Ingresar numero: 540
ver imagen...

            left_row    =  100
            right_row   =  440
            left_col    =  120
            right_col   =  540
```

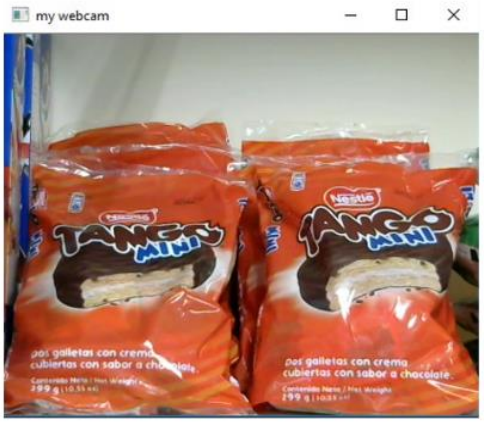


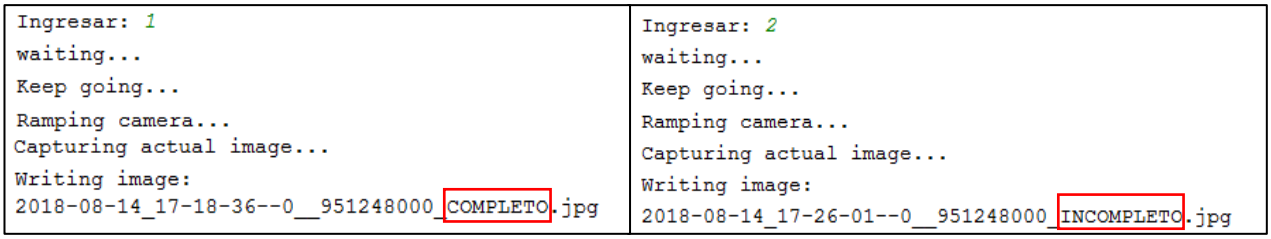
Figura 4.4: Preprocesamiento en la visión de la cámara para adquisición de datos.

```
========================= SAMPLING  ==========================
ANALYSIS OF DATASET: sampling/set_train/
TOTAL FILES: 4325
PRODUCTOS:
    %TOTAL      SAMP       COM    ...     %INT_INC    V         ID
0     6.0%  241/4325         -    ...            -  241  000000000
1    10.0%  432/4325   134/432    ...        0.69%    -  145263000
2     5.0%  195/4325    65/195    ...        0.67%    -  152634000
3     6.0%  249/4325    75/249    ...         0.7%    -  163593000
4    10.0%  429/4325   149/429    ...        0.65%    -  256348000
5     9.0%  409/4325   117/409    ...        0.71%    -  326598000
6    11.0%  495/4325   178/495    ...        0.64%    -  426895000
7    10.0%  441/4325   127/441    ...        0.71%    -  761349000
8    13.0%  552/4325   196/552    ...        0.64%    -  854269000
9    10.0%  430/4325   110/430    ...        0.74%    -  926583000
10   10.0%  452/4325   167/452    ...        0.63%    -  951248000

[11 rows x 8 columns]
1.- Tomar otra foto completa de producto PAQUETE DE TANGOS GRANDE
2.- Tomar otra foto incompleta de producto PAQUETE DE TANGOS GRANDE
3.- Salir
Ingresar:
```

Figura 4.3: Menú de selección de adquisición de datos.

Al seleccionar una de las opciones para tomar datos, se inicia el proceso de fondo para cortado, redimensionado y guardado en directorios específicos. Se notará en la Figura 4.5.a que, si se toma una foto con estado completo, este estado será parte del etiquetado del dato. Así mismo en la Figura 4.5.b se distingue el estado incompleto.



(a) Toma de foto con esta completo.

(b) Toma de foto con esta incompleto.

Figura 4.5: Muestra de línea de comandos de adquisición de datos para cada estado.

Cabe indicar que la etiqueta de los datos adquiridos (fotos) se emplea de la siguiente manera FECHA__IDPRODUCTO_ESTADO.jpg

4.1.4 Resultados de la adquisición de datos

A continuación, se presentan los resultados finales de los conjuntos de datos de entrenamiento (Figura 4.6), de validación (Figura 4.7) y de prueba (Figura 4.8).

ANALYSIS OF DATASET: ../sampling/set_train/
TOTAL FILES: 4326
PRODUCTOS:

	%TOTAL	SAMP	COM	%INT_COM	INC	%INT_INC	V	ID	PROD
0	6.0%	241/4326	-	-	-	-	241	000000000	NO PRODUCTO
1	10.0%	432/4326	134/432	0.31%	298/432	0.69%	-	145263000	DESINFECTANTE
2	5.0%	195/4326	65/195	0.33%	130/195	0.67%	-	152634000	CEREAL KELLOG GRANDE
3	6.0%	249/4326	75/249	0.3%	174/249	0.7%	-	163593000	GALLETAS AMOR GRANDE
4	10.0%	429/4326	149/429	0.35%	280/429	0.65%	-	256348000	FUNDA DE ARROZ GRANDE
5	9.0%	409/4326	117/409	0.29%	292/409	0.71%	-	326598000	PAPEL HIGENICO
6	11.0%	495/4326	178/495	0.36%	317/495	0.64%	-	426895000	LECHE TONI GRANDE
7	10.0%	441/4326	127/441	0.29%	314/441	0.71%	-	761349000	COLA 3L
8	13.0%	552/4326	196/552	0.36%	356/552	0.64%	-	854269000	ACEITE
9	10.0%	430/4326	110/430	0.26%	320/430	0.74%	-	926583000	BARRA DE JABON
10	10.0%	453/4326	167/453	0.37%	286/453	0.63%	-	951248000	PAQUETE DE TANGOS GRANDE

Figura 4.6: Resultados finales del conjunto de datos de entrenamiento.

```
ANALYSIS OF DATASET: ../sampling/set_test/
TOTAL FILES: 42
PRODUCTOS:
    %TOTAL   SAMP   COM %INT_COM   INC %INT_INC  V         ID                      PROD
0     7.0%   3/42     -        -     -        -  3  000000000               NO PRODUCTO
1    10.0%   4/42   2/4     0.5%   2/4     0.5%  -  145263000               DESINFECTANTE
2     7.0%   3/42   2/3    0.67%   1/3    0.33%  -  152634000      CEREAL KELLOG GRANDE
3     7.0%   3/42   1/3    0.33%   2/3    0.67%  -  163593000      GALLETAS AMOR GRANDE
4    10.0%   4/42   2/4     0.5%   2/4     0.5%  -  256348000     FUNDA DE ARROZ GRANDE
5    10.0%   4/42   2/4     0.5%   2/4     0.5%  -  326598000            PAPEL HIGENICO
6    10.0%   4/42   3/4    0.75%   1/4    0.25%  -  426895000         LECHE TONI GRANDE
7    10.0%   4/42   2/4     0.5%   2/4     0.5%  -  761349000                  COLA 3L
8    12.0%   5/42   2/5     0.4%   3/5     0.6%  -  854269000                   ACEITE
9    10.0%   4/42   1/4    0.25%   3/4    0.75%  -  926583000            BARRA DE JABON
10   10.0%   4/42   2/4     0.5%   2/4     0.5%  -  951248000  PAQUETE DE TANGOS GRANDE
```

Figura 4.8: Resultados finales del conjunto de datos de pruebas.

```
ANALYSIS OF DATASET: ../sampling/set_validation/
TOTAL FILES: 127
PRODUCTOS:
    %TOTAL    SAMP   COM %INT_COM    INC %INT_INC   V         ID                      PROD
0    16.0%  20/127     -        -      -        -  20  000000000               NO PRODUCTO
1     6.0%   7/127   4/7    0.57%    3/7    0.43%   -  145263000               DESINFECTANTE
2    15.0%  19/127  8/19    0.42%  11/19    0.58%   -  152634000      CEREAL KELLOG GRANDE
3    13.0%  17/127  6/17    0.35%  11/17    0.65%   -  163593000      GALLETAS AMOR GRANDE
4    13.0%  16/127  6/16    0.38%  10/16    0.62%   -  256348000     FUNDA DE ARROZ GRANDE
5     4.0%   5/127   1/5     0.2%    4/5     0.8%   -  326598000            PAPEL HIGENICO
6     5.0%   6/127   2/6    0.33%    4/6    0.67%   -  426895000         LECHE TONI GRANDE
7     7.0%   9/127   3/9    0.33%    6/9    0.67%   -  761349000                  COLA 3L
8     3.0%   4/127   2/4     0.5%    2/4     0.5%   -  854269000                   ACEITE
9     9.0%  12/127  4/12    0.33%   8/12    0.67%   -  926583000            BARRA DE JABON
10    9.0%  12/127  4/12    0.33%   8/12    0.67%   -  951248000  PAQUETE DE TANGOS GRANDE
```

Figura 4.7: Resultados finales del conjunto de datos de validación.

Se nota que algunos productos tienen menos cantidad de datos que otros pues el espacio de posibles escenarios es menor. También se nota que el producto de índice 0 (cero) se refiere a NO PRODUCTO, que es el producto ideado para señalas a un frente de lineal vacío.

4.2 Proceso de prueba de las redes neuronales convolucionales entrenadas realizando inferencias

Para validar los resultados del entrenamiento con el conjunto de datos adquirido, se realizó una prueba con el conjunto de datos de prueba que contiene 42 imágenes distintas. Se creó un menú para realizar pruebas de manera general entre todos los modelos entrenados. Se presenta a continuación una sesión de prueba de este algoritmo con la red neuronal Xception.

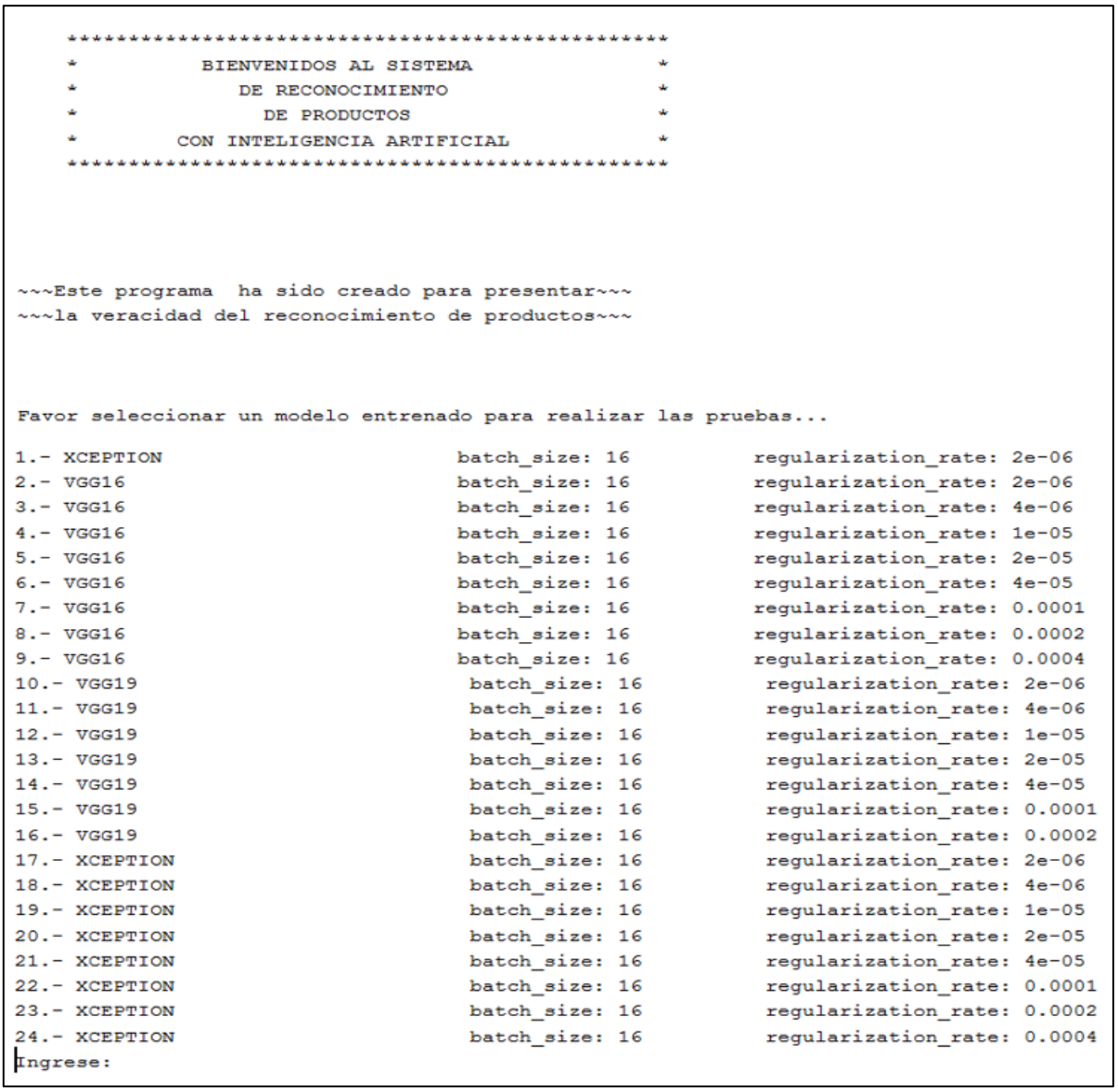


Figura 4.9: Menú de selección de red neuronal.

En primera instancia se tiene un menú de selección de modelo como se aprecia en la Figura 4.9, una vez seleccionada la red se realiza la carga de pesos a la red neuronal. Es importante comentar que la red no puede variar en ningún aspecto. Si se aumenta un bloque más o se elimina, el modelo no podrá cargarse normalmente para los pesos del entrenamiento que se realizó.

En la parte del proceso de pruebas que se visualiza en la Figura 4.10 se presenta la selección de validar una sola imagen y de validar sobre todo el conjunto de prueba. Para obtener resultados completos se selecciona la opción 2, la opción 1 se utilizó para presentar el resultado de uno de los datos del conjunto de pruebas e inferir sobre su éxito o fallo.

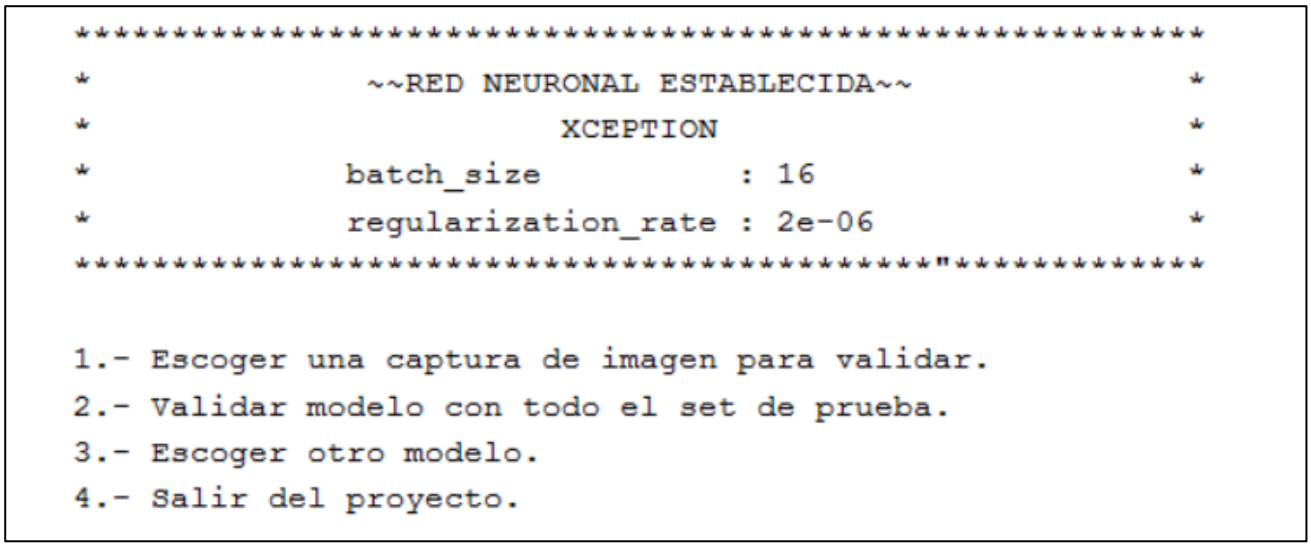


Figura 4.10: Opciones relevantes del menú de pruebas.

En la Figura 4.11 observa que la selección de la foto tuvo una inferencia correcta, por tanto, presenta el mensaje de correcto; mientras que en la Figura 4.12 se notará una inferencia incorrecta. También se visualiza los mensajes indicados en el alcance de proyecto como reporte de la inferencia, pero solamente para las inferencias correctas.

```
**********************************************************
La inferencia de la red es:
CEREAL KELLOG GRANDE            COMPLETO
**********************************************************
INFERENCIA REALIZADA FUE:       CORRECTA
**********************************************************
REPORTE DE INFERENCIA:
PRODUCTO CORRECTO
**********************************************************
NOTA:
        El producto CEREAL KELLOG GRANDE
        se encuentra en optimas condiciones.
**********************************************************
```

Figura 4.11: Prueba de datos realizada con inferencia correcta.

```
**********************************************************
La inferencia de la red es:
CEREAL KELLOG GRANDE            INCOMPLETO
**********************************************************
INFERENCIA REALIZADA FUE:       INCORRECTA
La inferencia correcta seria:
BARRA DE JABON                  INCOMPLETO
**********************************************************
REPORTE DE INFERENCIA:
        NO SE PRESENTARA REPORTE
        DE INFERENCIA ERRONEA

**********************************************************
```

Figura 4.12: Prueba de dato realizada con inferencia incorrecta.

4.3 Comparación entre diferentes modelos de redes neuronales.

Al entrenar las redes, se notó que este proceso demoraba mucho, y sus resultados no eran muy favorables; esto se dio en consecuencia de iniciar el entrenamiento sin pesos haciendo que estos sean iniciados aleatorios y que su ajuste sea más exhaustivo. Para solucionar este problema y dado que la cantidad de datos no se podía aumentar considerablemente, se utilizó pesos de entrenamiento para cada red neuronal sobre el conjunto de datos ImageNet [23]. ImageNet es un conjunto de datos reconocido en competencias para reconocimiento de objetos, su tamaño es masivo y es de libre uso.

La implementación en Keras de las redes neuronales escogidas permite indicar si se desea o no descargar y utilizar el archivo de pesos (con extensión .h5) de una red con ImageNet.

En la Tabla 4.1 a continuación se presentan valores de entrenamientos realizados y contrastados con los datos del conjunto de pruebas.

Tabla 4.1: Resultados de pruebas sobre conjunto de datos de prueba.

Tipo de Red Neuronal	Éxitos sobre 42	EPOCHS	Regularization rate	val_Output-Product_Estate_acc	val_Output-Product_ID_acc	Errores	
						ID	EST
RESNET50	42	10	2.00E-05	0.984	1.00		
XCEPTION	42	8	4.00E-05	0.964	1.00		
XCEPTION	42	9	2.00E-05	0.929	1.00		
INCEPTIONV3	41	8	2.00E-04	0.992	1.00		
DENSENET169	41	8	4.00E-05	0.984	1.00	x	x
RESNET50	41	8	1.00E-04	0.968	1.00		x
RESNET50	41	4	2.00E-04	0.960	1.00		x
RESNET50	41	4	4.00E-05	0.952	1.00		x
INCEPTIONV3-RESNETV2	40	5	4.00E-06	0.968	1.00		x
							x

Todas las redes mencionadas en la etapa 3 de la metodología de este proyecto fueron entrenadas y probadas bajo los mismos conjuntos de datos de entrenamiento, validación y prueba.

De la tabla xx podemos notar que los modelos escogidos finalmente fueron aquellos que en prueba dieron un resultado superior a 90% de acierto. De entre los modelos presentes, tres de ellos cumplen con un 100% de éxito de prueba, por tanto, verificamos su porcentaje de clasificación producto y estado de producto durante el entrenamiento y de esto notamos un porcentaje de 100% para la tarea de clasificar el producto y un porcentaje mayor al 90% en la tarea de clasificación o reconocimiento de estado.

Al ser muy similares en entrenamiento y con 100% de éxito en pruebas, se escoge de entre estos al tipo de red Xception con radio de regularización (variable usada como parte del inicializador de kernel) de valor 4.00E-05 por necesitar menor número de repeticiones para ser entrenado totalmente, esto será de ayuda para el reentrenamiento cuando sea requerido.

4.3.1 Diseño de red neuronal escogida

Para este proyecto, que tiene como propósito la multiclasificación de objetos, se decidió utilizar como base la red neuronal para clasificación de objetos desarrollada por Google, denominada ”Xception” [13].

Este modelo viene precargado en el módulo Keras de Python junto a su archivo de preentrenamiento sobre el conjunto de imágenes denominada ”ImageNet” [23]. La arquitectura de esta red neuronal puede ser observada en la Figura 4.13.

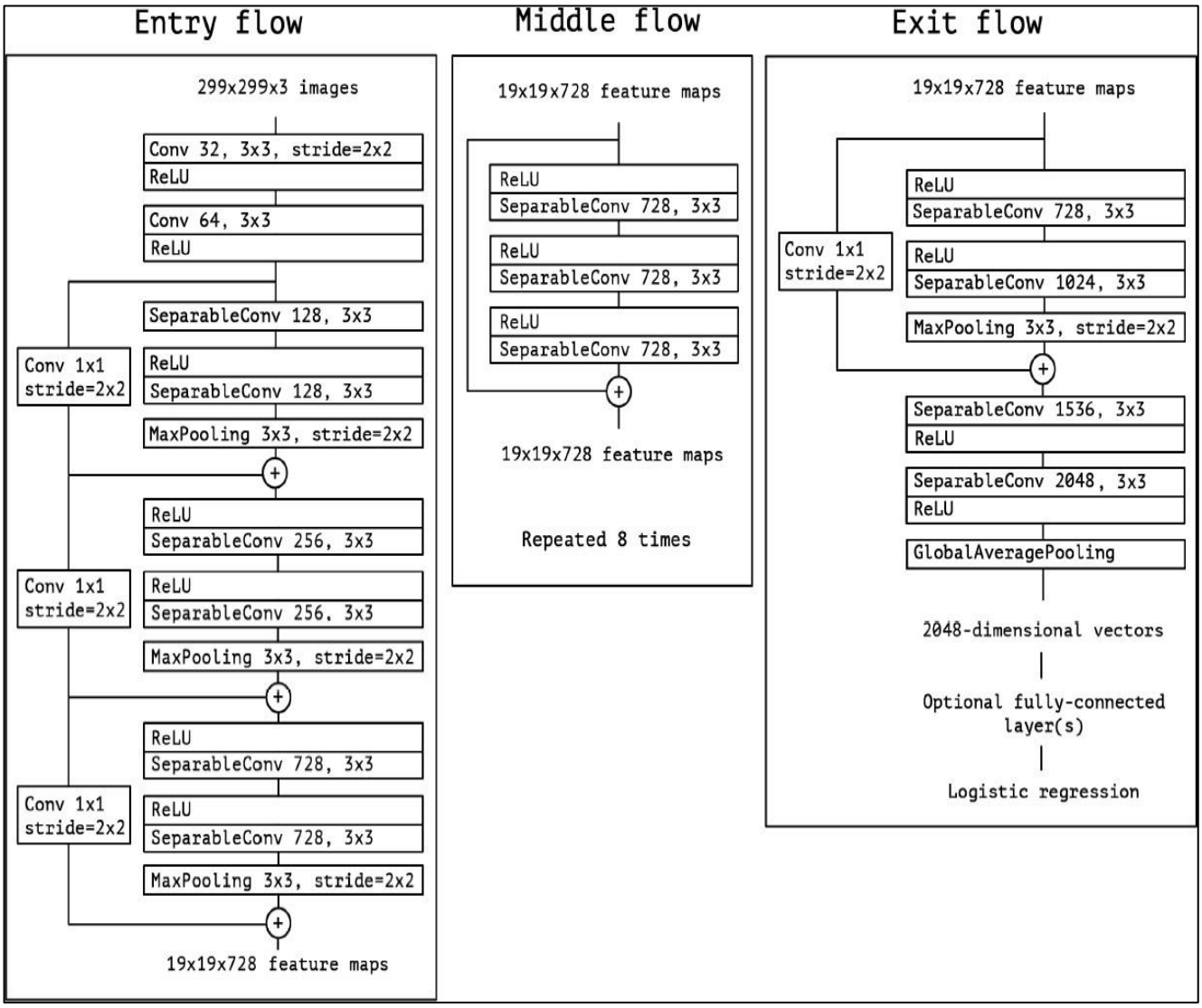


Figura 4.13: Arquitectura de la red Xception. [25]

La red neuronal artificial Xception de Keras incorpora algunos parámetros que pueden ser modificados. Entre ellos se detallan los más importantes:

- include_top: parámetro usado para indicar si la red Xception deberá incluir las capas de salida. Para el proyecto se indicó este parámetro como FALSO, pues en el modelamiento se diseñará la salida personalizada a las necesidades de este.

- weights: parámetro para incluir preentrenamiento. Se indicó como ”imagenet” para utilizar los pesos de esta red preentrenados en la base de imágenes de imagenet.

Finalmente, para brindar las salidas necesarias (producto y estado), se añade a la salida de Xception otras capas que se pueden visualizar en la tercera y cuarta jerarquía en orden de arriba hacia abajo de la .Figura 4.14

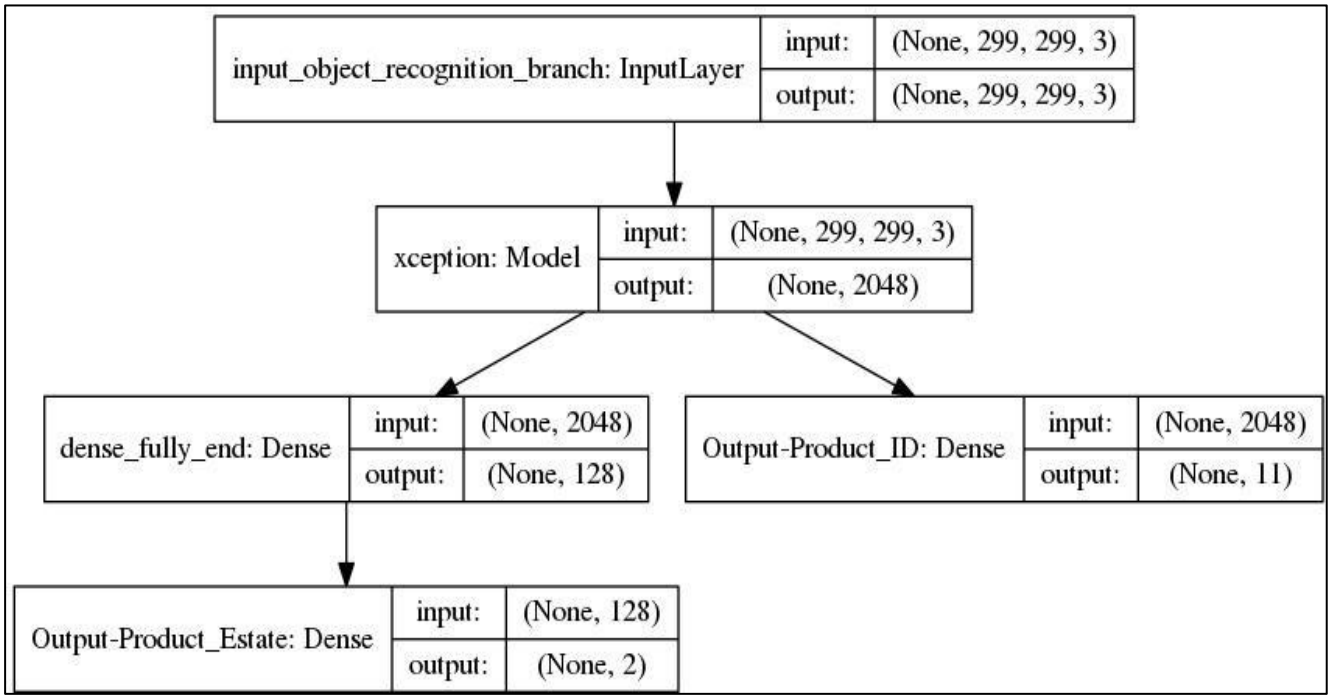


Figura 4.14: Arquitectura usada en el modelo del proyecto.

Se presenta en la Figura 4.15 el monitoreo de las curvas de aprendizaje de las variables entrenadas más relevantes de la red Xception escogida. Los datos para las gráficas fueron extraídos durante el entrenamiento por cada ciclo de entrenado.

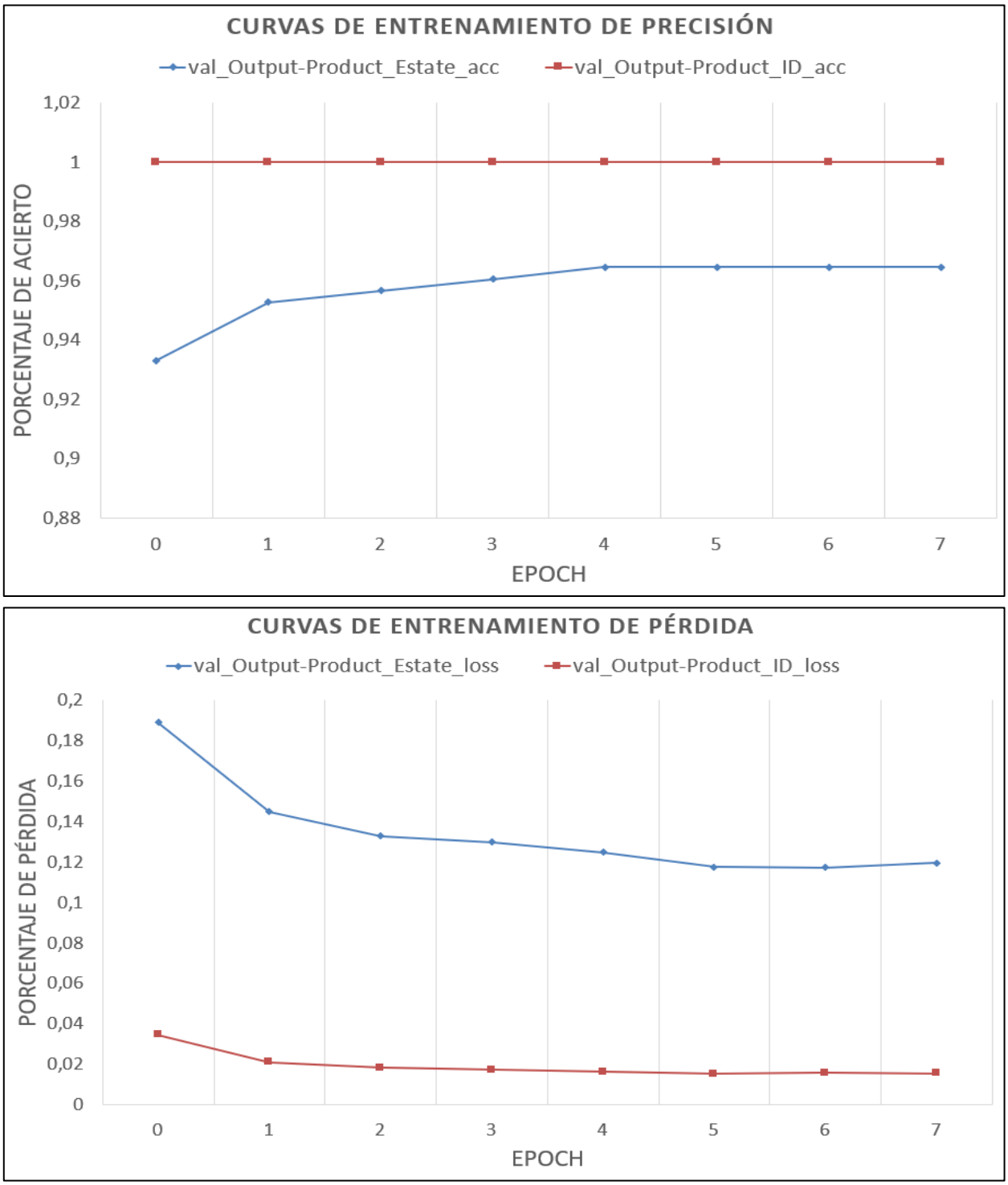


Figura 4.15: Curvas de variables entrenadas monitoreadas.

CAPÍTULO 4

5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Se pudo comprobar la efectividad de las redes neuronales convolucionales para solucionar el problema de multiclasificación de objetos, específicamente para los diez productos escogidos de un supermercado, las estadísticas de clasificación se evidencias en la Tabla 4.1, donde la mayoría de RNC´s son efectivas reconociendo los productos en el ambiente de lineal de góndola esquematizado.

- Al entrenar una red neuronal desde cero los pesos y desviamientos son inicializados aleatoriamente haciendo que la red tarde más en entrenarse; iniciar una red neuronal con su propio archivo de preentrenamiento con bases de datos grandes (para este caso de imágenes) beneficiará al entrenamiento haciendo este proceso menos exhaustivo.

- Se obtuvo un resultado de 100% de pruebas en los productos clasificados correctamente con el diseño a partir de la arquitectura de la red neuronal convolucional Xception.

5.2 Recomendaciones

- Implementar sensores de proximidad en el producto final para reducir al máximo posible el error de falsos estados incompletos que se pueden presentar en las inferencias de productos que solo tienen una pieza de frente de lineal en la góndola.

- Para realizar de manera más automática la adquisición de los datos de entrenamiento y los datos de evaluación en caliente debe aplicarse técnicas de “Detección de Objetos”. La finalidad de esto es poder detectar los flejes (etiquetas) en la góndola y realizar un preproceso de la imagen menos manual.

- Realizar pruebas en locales de El Supermercado que mantengan buena luminosidad para que los datos obtenidos sean más fiables,

BIBLIOGRAFÍA

[1] B. T. Calle, Interviewee, *Servicio de Atención al.* [Entrevista]. 1950.

[2] F. Chollet, de *Deep Learning with Python*, Manning Publications, 2018, p. 4.

[3] «Logicalis,» LogicalisES, 26 2 2018. [En línea]. Available: https://blog.es.logicalis.com/analytics/inteligencia-artificial-el-tiempo-es-ahora.

[4] A. M. Turing, «Computing Machinery and Intelligence,» 1950.

[5] J. P. a. A. Gibson, de *Deep Learning: A Practitione's Approach 1st Ed.*, O'Reilly Media., 2017, pp. 23-25.

[6] W. M. y. W. Pitts, «A Logical Calculus of the Ideas Immanent in Nervous Activity,» 1943.

[7] D. Julian, «Concepto de Redes Neuronales Artificiales,» de *Designing Machine Learning Systems with Python*, Packt Publishing Ltd., 2016, pp. 128 - 136.

[8] D. Julian, «Implementación de una red neuronal Artificial,» de *Designing Machine Learning Systems with Python*, Pack Publishing Ltd., 2016, pp. 139 - 147.

[9] Q. G. a. T. E. L.-T. L.S. S. J. A. Xinyu Chen, «In situ tensorview: In situ visualization of convolutional neuronal networks,» 2018.

[10] M. Eickenberg, «Seeing it all: Convolutional Network layers map the function of the human visual system.,» 2017.

[11] N. Kalchbrenner, «Neuronal machine translation in linear time,» 2016.

[12] C. N. d. S. a. M. Gatti, «Deep Convolutional Neuronal Networks for Sentiment Analysis of Short Texts,» 2014.

[13] F. Chollet, «Xception: Deep Learning with Depthwise separable convolutions,» 2017.

[14] DIARLU, *Lenguajes de Programación más utilizados del 2018,* 2018.

[15] P. Documentation, *Aplicaciones para el lenguaje de programación python..*

[16] A. A. P. B. E. B. Z. C. C. C. G. S. C. A. D. J. D. M. D. S. G. I. G. A. H. G. I. M. I.-. a. Y. J. R. J. L. K. M. K. J. L. D. M. R. M. S. M. M. Abadi, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems,* 2015.

[17] F. Chollet, *Keras,* 2015.

[18] G. H. a. Z. L. a. L. v. d. Maaten, «Densely Connected Convolutional Networks,» CoRR, 2016.

[19] C. S. and, «Rethinking the Inception Architecture for Computer Vision,» CoRR, 2015.

[20] C. S. and, «Inception-v4, Inception-ResNet and the Impact of Residual Connections,» 2016.

[21] K. S. and, «Very Deep Convolutional Networks for Large-Scale Image Recognition,» 2014.

[22] K. Y. a. W. X. a. Y. Gong, «Deep Learning with Kernel Regularization for Visual Recognition,» 2008.

[23] W. D. R. S. L. J. L. K. L. a. L. F.-F. J. Deng, *Imagenet; una base de datos de imágenes jerárquicas a gran escala,* 2009.

[24] F. Chollet, «Propiedad de Jerarquía de las redes neuronales convolucionales.,» de *Deep Learning with Python*, Manning Pulications., 2018, p. 123.

[25] F. Chollet, *Xception: Deep learning with depthwise separable convolutions. fig 5,* 2017.