# Desarrollo de una aplicación para la predicción de ingredientes y recetas de cocina por medio de TensorFlow y máquinas de soporte vectorial*

*Yeny Muñoz-Castaño*[**]
*Luis Castillo-Ossa*[***]
*Omar Castrillón-Gomez*[****]
*Felipe Buitrago-Carmona*[*****]
*Santiago Loaiza Giraldo*[******]

Recibido: 11/09/2019 • Aceptado: 13/07/2020
https://doi.org/10.22395/rium.v19n37a10

***Resumen***

Este artículo es derivado de un proyecto de investigación en el cual se realizó el desarrollo de una aplicación para la predicción de ingredientes y recetas por medio de TensorFlow y máquinas de soporte vectorial. Se realizó un esquema de la arquitectura general, luego se desarrolló una red neuronal y después se efectuó la ejecución de la máquina de soporte vectorial. Finalmente, se realizó la integración en una aplicación que permite al usuario seleccionar imágenes de los ingredientes para su predicción y de la receta de cocina para desayunos de manera didáctica. Se concluyó que el sistema tiene un promedio de precisión del 75,8 % para las 17 categorías de ingredientes y de 71 % para el clasificador de recetas. Adicionalmente, se realizó un análisis de esta estabilidad y se observó que todos los resultados eran iguales en términos estadísticos.

**Palabras clave**: reconocimiento; imagen; *TensorFlow*; SVM; redes neuronales.

* Articulo original derivado del proyecto de investigación "Prototipo computacional para la fusión y análisis de grandes volúmenes de datos en entornos IOT (internet de las cosas) a partir de técnicas de *machine learning* y arquitecturas seguras entre sensores, para caracterizar el comportamiento e interacción de los usuarios en un ecosistema de *connected home"* (código 36715), financiado por la Dirección de investigaciones de Manizales (DIMA) de la Universidad Nacional de Colombia (Sede Manizales). Periodo de ejecución: 2018 - 2019.

** Ingeniera electrónica. Miembro del Grupo de Investigación Innovación y Desarrollo Tecnológico. Universidad Nacional de Colombia, Sede Manizales. Correo electrónico: yymunozc@unal.edu.co. Orcid: 0000-0002-5853-2646.

*** Ph. D. Miembro del Grupo Gitir. Facultad de ingenierías, Universidad de Caldas. Correo electrónico: luis.castillo@ucaldas.edu.co. Orcid: 0000-0002-2878-8229.

**** Ph. D. Miembro del Grupo de Investigación Innovación y Desarrollo Tecnológico. Universidad Nacional de Colombia, Sede Manizales. Correo electrónico: odcastrillong@unal.edu.co. Orcid: 0000-0003-3713-0696.

***** Ingeniero de sistemas y computación. Miembro del grupo de investigación Semillero de Inteligencia Artificial I3A - Grupo Gitir. Universidad de Caldas. Correo electrónico: felipe.1701416667@ucaldas.edu.co. Orcid: 0000-0002-1085-0979.

****** Ingeniero de sistemas y computación. Miembro del grupo de investigación Semillero de Inteligencia Artificial I3A - Grupo Gitir. Universidad de Caldas. Correo electrónico: santiago.1701412598@ucaldas.edu.co. Orcid: 0000-0001-6672-9804.

# Development of an Application for the Prediction of Kitchen Ingredients and Recipes through TensorFlow and Support-Vector Machines

***Abstract***

This article is derived from a research project in which an application for the prediction of ingredients and recipes by TensorFlow and support-vector machines was developed. A scheme with general architecture was developed, then a neural network was implemented, and then, the support-vector machine was run. After that, they were integrated via an application that allows the user to select ingredients' images for their prediction and the prediction of kitchens recipe in a didactic manner. It was concluded that the system has an average precision value of 75.8% and 71% for 17 ingredients categories and recipes classifier. In addition, sensitivity testing was performed on the application resulting on statistically equivalent results.

**Keywords:** Recognition, image, TensorFlow, SVM, Neural Networks.

# Desenvolvimento de um aplicativo para a predição de ingredientes e receitas de cozinha por meio de TensorFlow e máquinas de suporte vetorial

***Resumo***

Este artigo é derivado de um projeto de pesquisa no qual se realizou o desenvolvimento de um aplicativo para a predição de ingredientes e receitas por meio de TensorFlow e máquinas de suporte vetorial. Realizou-se um esquema da arquitetura geral, desenvolveu-se uma rede neuronal e depois efetuou-se a execução da máquina de suporte vetorial. Finalmente, realizou-se a integração num aplicativo que permite ao usuário selecionar imagens dos ingredientes para sua predição e da receita de cozinha para cafés da manhã de maneira didática. Concluiu-se que o sistema tem uma média de precisão de 75,8 % para as 17 categorias de ingredientes e de 71 % para o classificador de receitas. Adicionalmente, realizou-se uma análise dessa estabilidade e observou-se que todos os resultados eram iguais em termos estatísticos.

**Palavras-chave**: reconhecimento; imagem; TensorFlow; SVM; redes neuronais.

## INTRODUCCIÓN

El reconocimiento de imágenes y objetos en nuestro entorno está en auge. Uno de los métodos más utilizados es el uso de redes neuronales, las cuales han logrado clasificar de 1,2 millones de imágenes con 1000 clases diferentes [1]; igualmente, se han utilizado para extracción de patrones característicos de imágenes [2]. Así mismo, se han implementado técnicas de *deep learning*, el cual permite que los modelos computacionales compuestos de múltiples capas de procesamiento aprendan representaciones de datos con múltiples niveles de abstracción [3]. En la misma línea, se han implementado modelos como *Bag of Words* a fin de realizar un reconocimiento con base en el contenido de las imágenes [4]; o filtros como el de Base Haar y clasificadores en cascada para la extracción de características sobre imágenes digitales, con los que se han obtenido porcentajes de detección de rostro y de ojos de 100 % y 92 %, respectivamente [5].

Las máquinas de soporte vectorial (*support-vector machines,* SVM) constituyen otra de las técnicas utilizadas a este respecto: aprenden a predecir una nueva clase de muestra a partir de ejemplos. Esta técnica está basada en la minimización de riesgo estructural (*structural risk minimization,* SRM) [6] la cual se ha utilizado para la clasificación automática de señales sísmicas [7] dentro de la identificación de fracturas naturales en pozos de yacimientos de hidrocarburos: su exactitud osciló entre 72,3 % y 82,2 % en cinco pozos evaluados del campo estudiado [8]. También se la ha usado para el reconocimiento de patrones mediante huellas dactilares de descargas parciales con filtrado de *wavelet,* con altas tasas de reconocimiento [9].

Sumado a lo anterior, se han implementado SVM junto con los métodos de Kernel para mejorar la inferencia transductiva de motores de búsqueda de texto [10]; y también se han empleado para el reconocimiento de voz, en aras de emular la toma de decisiones de los humanos (tarea de clasificación y análisis de audio realizada por los técnicos): los resultados del correspondiente estudio arrojaron un porcentaje de clasificación exitosa en comparación con una plataforma automática llamada *CheckMyRoutes* [11].

TensorFlow ha sido utilizado para la construcción de una API (*Application Programming Interface*) de Python denominada *InferPy* para el modelado probabilístico construido sobre Edward y TensorFlow [12]. Así mismo, se ha usado para acelerar la inversión sísmica geoestadística, la cual se puede incluso mejorar el modelo utilizando *graphics processing unit*s (GPU) [13]. También se han realizado simulaciones con *CitySim*, un simulador que combina la energía de construcción y TensorFlow, el cual permite investigar algoritmos para el ahorro de energía y respuesta a la demanda [14]. Se han usado, además, *deep convolutional generative adversarial networks* (DCGAN), es decir, redes de confrontación generativas de convolución profunda: se ha realizado una exploración de esta última a través de TensorFlow y se ha encontrado

que el modelo DCGAN mejora de forma significativa frente al modelo de generación de caras virtuales [15].

El uso de las herramientas citadas ha empezado a difundirse en empresas para optimizar sus procesos dado que son versátiles y mejoran el nivel de desarrollo. Así mismo, miles de startups e investigadores universitarios apuestan por TensorFlow como base para desarrollar sus propios modelos de inteligencia artificial [16]. Por lo tanto, es importante desde la academia incentivar el uso de estas herramientas con miras a desarrollar aplicaciones que puedan ser útiles a la sociedad y, en esta línea, que sean usadas en entornos de gran importancia en la cotidianidad, como la cocina. A este respecto, si bien en la actualidad se ha automatizado el hogar con sistemas de iluminación inteligente, calefacción y ventilación enfocados en la reducción del consumo de energía [17], es necesario hacer un mayor aporte tecnológico con el uso de estas técnicas nuevas para que la automatización mejore y, con ello, sea mucho más eficaz.

Con lo anterior, el objetivo de esta investigación fue desarrollar una aplicación para la predicción de ingredientes y recetas por medio de TensorFlow y máquinas de soporte vectorial. Para esto se realizó un esquema de la arquitectura general, luego se desarrolló una red neuronal y después se efectuó la ejecución de la máquina de soporte vectorial. Finalmente se realizó la integración en una aplicación que permitiría al usuario seleccionar imágenes de los ingredientes para la predicción de los ingredientes y la receta de manera didáctica.

Este desarrollo, realizado con TensorFlow y máquinas de soporte vectorial para uso en la cocina, permitiría reconocer los ingredientes y realizar una clasificación de recetas que se tienen en una cocina. Aunque en la fase de la investigación que de describe aquí se realizaron las pruebas con imágenes tomadas de Google (210 por cada categoría de ingredientes), se espera que en la fase posterior, de entrenamiento, el sistema se implemente en un microcontrolador de bajo consumo como Raspberry Pi 3, dado que ya se ha usado con anterioridad para supervisar y controlar procesos a nivel industrial [18]; igualmente, se aspira a incrementar el número de ingredientes reconocidos (100) y el reconocimiento en recetas (20), y establecer una asociación con un servidor IoT, dado que permite la gestión de entornos domésticos [19].

## 1. MATERIALES Y MÉTODOS

Esta sección presenta la información y las descripciones cortas de los procedimientos realizados para desarrollar de una aplicación para la predicción de ingredientes y recetas por medio de TensorFlow y máquinas de soporte vectorial.

### 1.1 Paso 1: descripción de la arquitectura general

Se procedió a desarrollar un sistema con el modelo cliente - servidor para obtener la predicción de 17 categorías de ingredientes y 6 categorías de recetas (tabla 1). La parte denominada 'cliente', esto es, aquello conocido como *front-end*, que brindaría al usuario final una relación agradable e intuitiva con el sistema, se desarrollaría en código HTML con JavaScript para vincularla con la parte de servidor (*back-end*). En la figura 1 se observa la arquitectura mínima planteada.

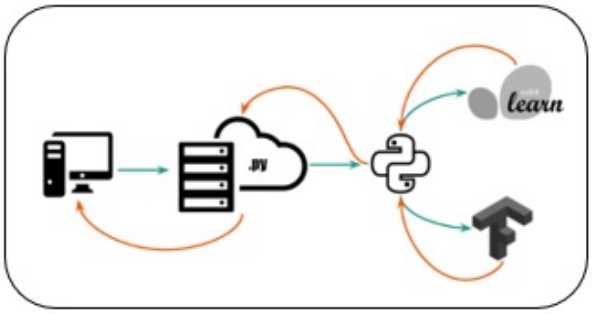


Figura 1. Descripción de la arquitectura general

Fuente: elaboración propia.

Tabla 1. Categorías generales

| Ítem | Descripción |
|---|---|
| Categoría A: ingredientes | Huevos, arepas, mantequilla, chocolate, pan, cereales, café, leche, tocino, changua, tamal, papas, calentado, yuca frita, jugo de naranja, yogur, pollo. |
| Categoría B: recetas de desayunos | Paisa, cafetero, rolo, fitness, tolimense, americano. |

Fuente: elaboración propia.

### 1.2 Paso 2: desarrollo de la red neuronal

El modelo de las redes neuronales surgió de simulaciones abstractas de sistemas nerviosos biológicos, constituidos a su vez por neuronas o nodos interconectados entre sí [20]. En un modelo de neurona artificial se tienen cuatro componentes básicos: primero, un conjunto de conexiones o sinapsis que determinan el comportamiento de la neurona; segundo, un sumador que se encarga de unir todas las entradas multiplicadas por las respectivas sinapsis; tercero, una función de activación no lineal para limitar la amplitud de la salida de la neurona; y cuatro, un umbral exterior que determina cuándo se activa la neurona. En la figura 2 se observa el modelo [21].

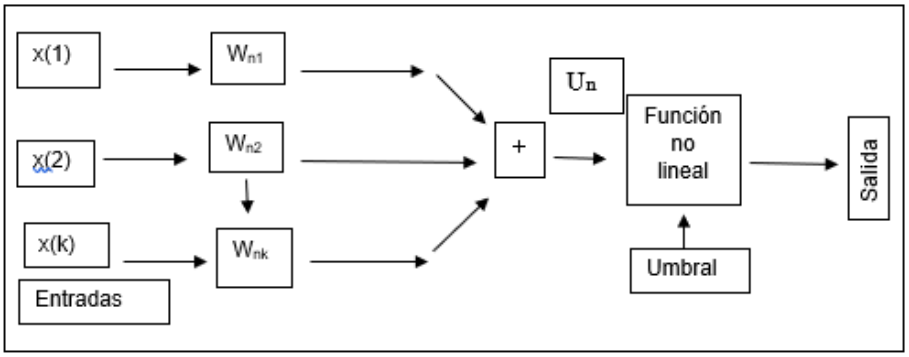


Figura 2. Modelo de red neuronal

Fuente: adaptada de [21].

En la ecuación (1) se observa la operación matemática a realizar [21, p.15]

$$"U_{n=}\sum_{j=1}^{k} W_{nj}.x(j)" \tag{1}$$

Y en la ecuación (2) se observa la salida [21, p.15]

$$"salida = \rho(Un - umbral)" \tag{2}$$

Donde ρ es una función de activación no lineal; adicionalmente, se asocia el umbral a la salida *Un* mediante una entrada que vale -1 y un peso asociado. Por lo tanto, en las ecuaciones (3) y (4), se observa el cambio del umbral [21, p.15]

$$"umbral = -W_{no} \rightarrow \{U_{n=}\sum_{j=1}^{k} W_{nj}.x(j) \rightarrow salida = \rho(U_n)" \tag{3}$$

O

$$"umbral = -W_{no} \rightarrow \{x(0) = 1 \rightarrow salida = \rho(U_n)" \tag{4}$$

Este modelo planteado es el más común. Empero, hay otros que no realizan un promedio de las entradas de forma directa, sino que realizan una transformación a dichas entradas —cuadrática, polinómica, esférica—; adicionalmente, también se podría tener consideración de las entradas anteriores [21].

El desarrollo de una red neural es el núcleo principal para el funcionamiento del sistema y se busca que brinde inteligencia específica al decidir sobre una imagen que está llegando como entrada de datos. Para ella se tuvieron en consideración los siguientes elementos:

- *Modelo:* la red neuronal diseñada es de tipo convolucional, muy similar a un perceptrón multicapa. Este se compone de una capa de entrada, una de salida y una o más que están ocultas; la figura 3 muestra un perceptrón típico [20]. En el desarrollo de esta investigación se definió un modelo de dos capas de convolución, las cuales podrán variar internamente o aumentar en cantidad por temas de rendimiento y porcentaje de precisión en versiones futuras del sistema. Ambas capas de convolución utilizan dos dimensiones con el objetivo, ya que la teoría las propone como ideales para el trabajo con imágenes; se utilizan cinco unidades como dimensión de la ventana de convolución (alto y ancho) y un paso por esta ventana. La primera capa cuenta con una salida de 16 filtros; y la segunda, con una de 36. A ellas se les aplica un proceso de discretización basado en muestras conocido como '*Max-pooling'* encaminado a reducir la muestra de una representación de entrada (imagen, matriz de salida de capa oculta, etc.) de tal suerte que se reduzca su dimensionalidad y se posibiliten suposiciones sobre las características contenidas en las subregiones agrupadas.

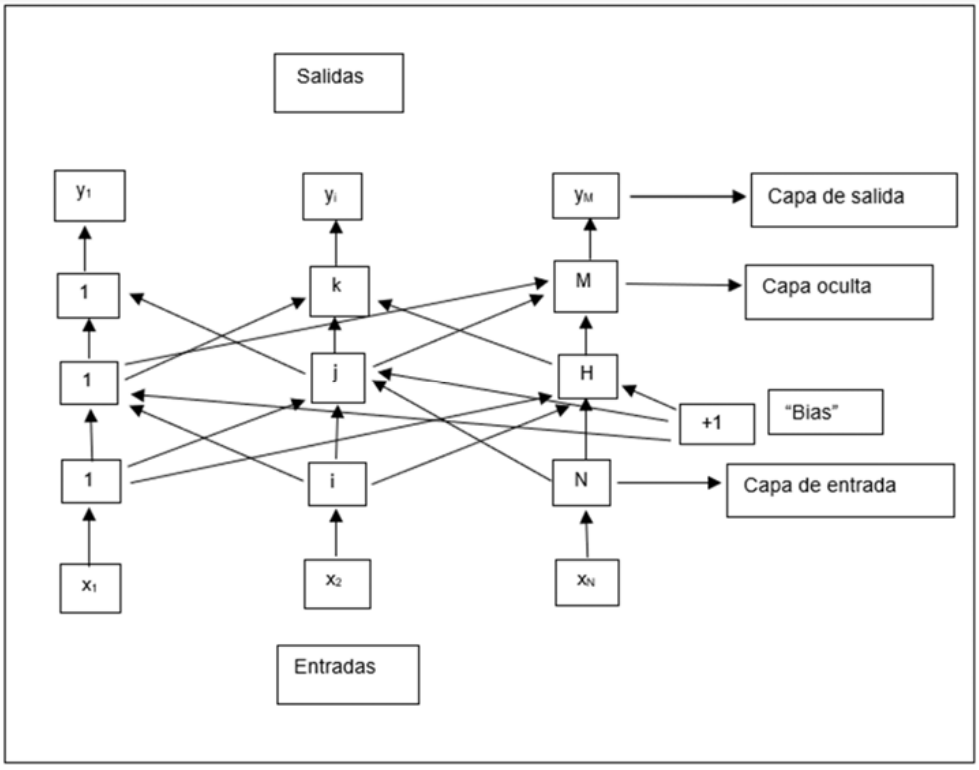


Figura 3. Modelo de perceptrón típico

Fuente: adaptada de [20].

- *Función de costo:* esta trata de determinar el error entre los valores estimado y real con el fin de optimizar los parámetros de la red neuronal. Para esta investigación se utilizó el estimador de raíz cuadrada media RMSE (ecuación 5): esta es una medida de precisión calculada como la raíz cuadrada media de los residuos, entendidos estos últimos como la diferencia entre los valores previsto (correcto) y real obtenido [22, p. 1].

$$"RMSE = \sqrt{\sum_{i=1}^{n} \frac{\left(\hat{y}_i - y_i\right)^2}{n}}" \tag{5}$$

- *Factor de aprendizaje:* el aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante la etapa de aprendizaje se reducen a la destrucción (el peso de la conexión toma el valor 0), modificación y creación (el peso de la conexión toma un valor distinto de 0) de conexiones entre las neuronas.

Podemos considerar que el proceso de aprendizaje ha terminado (ecuación 6) cuando los valores de los pesos permanecen estables. [23, p.1]

$$"\frac{dW_j}{dt} = 0" \tag{6}$$

En esta investigación se realizó un aprendizaje supervisado: se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor comprueba la salida generada por el sistema y, en caso de que no coincida con la esperada, se procederá a modificar los pesos de las conexiones [23].

### 13. Paso 3: desarrollo de la máquina de soporte vectorial

La teoría de la SVM está basada en la idea de minimización de riego estructural (SRM). En muchas aplicaciones, las SVM han mostrado tener gran desempeño, incluso más que máquinas de aprendizaje tradicional como las redes neuronales; y han sido introducidas como herramientas poderosas para resolver problemas de clasificación [6]. Dentro de la tarea de clasificación, las SVM pertenecen a la categoría de los clasificadores lineales puesto que inducen separadores lineales o hiperplanos, ya sea en el espacio original de los ejemplos de entrada si estos son separables o cuasi separables (ruido), o en un espacio transformado (espacio de características) si los ejemplos no son separables linealmente [24].

Se define un conjunto (ecuación 7) para una clasificación binaria de ejemplos separables linealmente [24, p. 2]:

$$"S = \{(x_1, y_1); \ldots\ldots (x_n, y_n)\}" \tag{7}$$

Donde $xi \ R^d$ e $yi \in \{+1, -1\}$ [24, p. 2] se puede definir un hiperplano de separación (figura 4A) como una función lineal que es capaz de separar dicho conjunto (ecuación 8) sin error [24, p. 2].

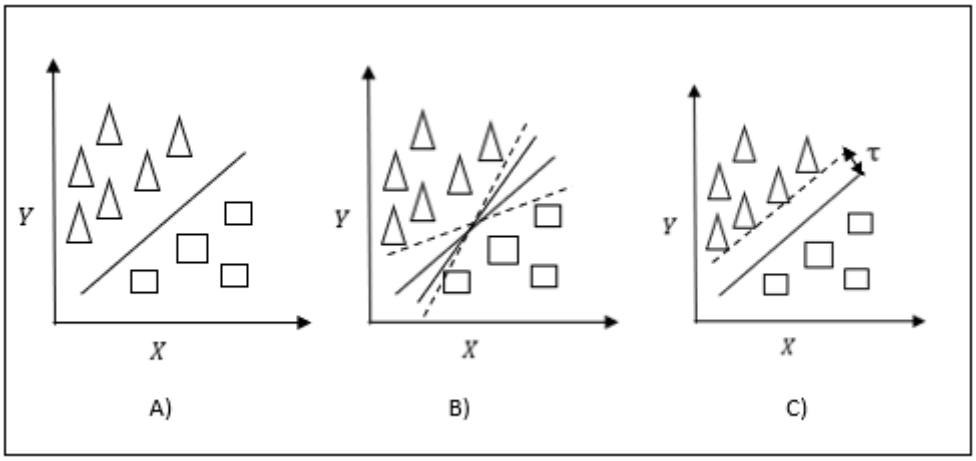


Figura 4. a) ejemplo de hiperplano de separación de dos clases; b) otros ejemplos de hiperplanos de separación; c) margen de un hiperplano de separación

Fuente: adaptada de [22].

$$"D(x) = (w_1, x_1), + \ldots + (w_d, x_d) + b = \langle w, x \rangle + b" \tag{8}$$

Donde w y b son coeficientes reales [24], el hiperplano de separación cumplirá las restricciones presentadas por las ecuaciones (9), (10), (11) y (12) para todo $x_i$ del conjunto de ejemplos [24, p. 3].

$$"\langle w, x_i \rangle + b \geq 0 \text{ sí } y_i = +1" \tag{9}$$

$$"\langle w, x_i \rangle + b \leq 0" \text{ si } "y_i = -1", "i = 1, \ldots.., n" \tag{10}$$

O también,

$$"y_i (<w, x_i> + b) \geq 0", "i = 1, \ldots\ldots., n" \tag{11}$$

O de forma más compacta,

$$"y_i D(x_i) \geq 0, i = 1, \ldots\ldots., n" \tag{12}$$

Ahora el hiperplano que permite separar los ejemplos no es único, como se observa en la figura 4B; por lo tanto, se define el concepto de *margen* en un hiperplano de separación denotado por τ como la mínima distancia entre dicho hiperplano y el ejemplo más cercano de cualquiera de las dos clases, como se observa en la figura 4C, si su margen de tamaño máximo se denominara óptimo.

Una propiedad del hiperplano de separación óptimo es que resulta equidistante respecto del ejemplo más cercano de cada clase: Por geometría (ecuación 13) se sabe que la distancia entre un hiperplano de separación $D(x)$ y un ejemplo $x'$ está dada por [24]:

$$\frac{\left|D\left(x'\right)\right|}{w} \quad (13)$$

ô

Al hacer uso de las ecuaciones (12) y (13), todos los ejemplos de entrenamiento (ecuación 14) cumplirán [24, p. 4].

$$\frac{y_i D(x_i)}{w} \geq \tau, \; i = 1, \ldots\ldots, n. \quad (14)$$

De la ecuación (13) se deduce que encontrar el hiperplano óptimo es igual a encontrar el valor de $w$ que maximiza el margen si se expresa la ecuación (14) de la siguiente manera [24, p. 4]:

$$y_i D\left(x_i\right) \geq w\tau, \; i = 1, \ldots\ldots, n. \quad (15)$$

Si se fija el producto τ y $w = 1$, se llega a la conclusión (ecuación 16) de que aumentar el margen es equivalente a disminuir la norma [24, p. 4].

$$\tau = \frac{1}{w} \quad (16)$$

En conclusión, un hiperplano optimo será aquel que posea un margen máximo y un valor mínimo de $\|w\|$.

Con todo lo dicho, y en términos más concretos, la SVM en el caso que nos ocupa es la agrupación de algoritmos que proporcionan los medios necesarios para elaborar procedimientos de aprendizaje supervisado; y su función es determinar, a partir de los ingredientes predeterminados y seleccionados por el usuario, a qué tipo de desayuno entre los establecidos podría pertenecer. Los elementos que componen una SVM son los siguientes:

- *Kernel:* Las funciones kernel permiten convertir lo que sería un problema de clasificación no lineal en uno de clasificación lineal con un espacio de dimensión mayor; normalmente son usadas en las SVM [25]. Para la instanciación del clasificador mencionado se definió el tipo de función bajo la cual se regiría. En este

caso puntual se utilizó el tipo *radial basis function* o función de base radial, definida a partir de pruebas y comparaciones con las demás opciones como la precisa para el *software* planteado.[26]

- *Grupo de datos:* para el caso del clasificador, la fuente de datos se genera en un archivo de valores separados por comas (*comma-separated values,* CSV); contiene, en una estructura definida por cada línea, un registro posible donde cada celda será un atributo en valor binario (1 pertenece, 0 no pertenece) y la última celda por cada línea representará la clase en un número entero asociado a un diccionario de datos.

### 1.4 Paso 4: pruebas de entrenamiento

Para el desarrollo de la red neuronal se realizó un módulo de pruebas: como entrada se proporciona una imagen; y con procedimientos de edición, almacenamiento, acceso y traspaso, esta llega como se desea a la red neuronal, la cual retornará entre ciertas categorías su decisión al respecto. Para el desarrollo de la máquina de soporte vectorial se realizó un módulo de pruebas cuya entrada correspondió a un arreglo con datos binarios por cada categoría (como se explicó): como resultado, el clasificador retornará un valor léxico de la clase asociada que representa una clase de desayuno definida, a la cual la receta podría pertenecer. Después de esto se realizan diversos procesos de entrenamiento en general, los cuales inician con muestras pequeñas (5 categorías, 10 imágenes por categoría), y luego se inician pruebas con los parámetros utilizados en la prueba de despegue con todo el grupo de datos (17 imágenes, 210 imágenes por categoría).

## 2. RESULTADOS

En primer lugar, se describirán aquí los resultados obtenidos en cuanto a los requerimientos: se observará un ejemplo de las categorías mencionadas, así como la descripción computacional de la arquitectura. En segundo lugar, se mostrarán resultados parciales de la red neuronal y las máquinas de soporte vectorial; y por último se presentará la interfaz para el usuario, así como las tablas de las predicciones obtenidas para ingredientes y recetas.

### 2.1 Pasos 1 y 2: descripción de la arquitectura general y desarrollo de la red neuronal

La programación de la arquitectura planteada se puede observar en el diagrama de la figura 5: allí se muestra que al interactuar con la interfaz, el usuario debe seguir un proceso en que ha de elegirse si se realiza el reconocimiento del ingrediente por medio de la red neuronal, o si se desea realizar la predicción de la receta a través de la SVM. Para el entrenamiento de la red neuronal, realizada en TensorFlow sobre Python, se

realiza un entrenamiento basado en épocas, cada una de las cuales tiene dos etapas: una de entrenamiento y una de pruebas. Como se estableció en secciones previas, para la red neuronal del sistema hay 210 imágenes por 17 categorías. A modo de ejemplo, en la tabla 2 se muestra una época del entrenamiento; en él, la cantidad es configurable. Finalizado esto, se obtiene un valor porcentual de efectividad con el set de datos del 87,06 %. La tabla 3 presenta los resultados del entrenamiento de la red neuronal con variaciones en los porcentajes de entrenamiento y validación.

Tabla 2. Programación, época d6e entrenamiento

```
python .\train.py
#Se enmarca el proceso de TensorFlow sobre PYTHON
Using TensorFlow backend.
Epoch 1/1
2890/2890 [==============================>] 2890/2890 [==============================>] - 6s 32ms/step -
loss: 3.4935 - acc: 0.0412
#ETAPA DE PRUEBA, SE EVALÚAN 680 IMÁGENES POR ÉPOCA (40 POR CADA CATEGORÍA)
680/680 [==============================>] 680/680 [==============================>] - 1s 9ms/step
#FINALIZADO EL PROCESO DE ENTRENAMIENTO, Y EN RELACIÓN CON LAS ETAPAS DE PRUEBA
#SE OBTIENE UN VALOR PORCENTUAL DE EFECTIVIDAD DE LA RED NEURONAL CON EL SET DE DATOS
acc: 87.06%
_________________________
Layer (type) Output Shape Param #
=================================================================
input_1 (InputLayer) (None, 16384) 0
_________________________
reshape_1 (Reshape) (None, 128, 128, 1) 0
_________________________
layer_conv1 (Conv2D) (None, 128, 128, 16) 416
_________________________
max_pooling2d_1 (MaxPooling2 (None, 64, 64, 16) 0
_________________________
layer_conv2 (Conv2D) (None, 64, 64, 36) 14436
_________________________
max_pooling2d_2 (MaxPooling2 (None, 32, 32, 36) 0
_________________________
flatten_1 (Flatten) (None, 36864) 0
_________________________
dense_1 (Dense) (None, 128) 4718720
_________________________
dense_2 (Dense) (None, 17) 2193
=================================================================
Total params: 4,735,765
Trainable params: 4,735,765
Non-trainable params: 0
```

Fuente: elaboración propia.

Tabla 3. Resultados del entrenamiento de la red neuronal para predicción de ingredientes con variaciones en los porcentajes de entrenamiento y validación

| Porcentaje de entrenamiento | Porcentaje de validación | Resultados |
|---|---|---|
| 20 % | 80 % | 40,7 % |
| 40 % | 60 % | 62,1 % |
| 60 % | 40 % | 73,3 % |
| 80 % | 20 % | 75,2 % |

Fuente: elaboración propia.

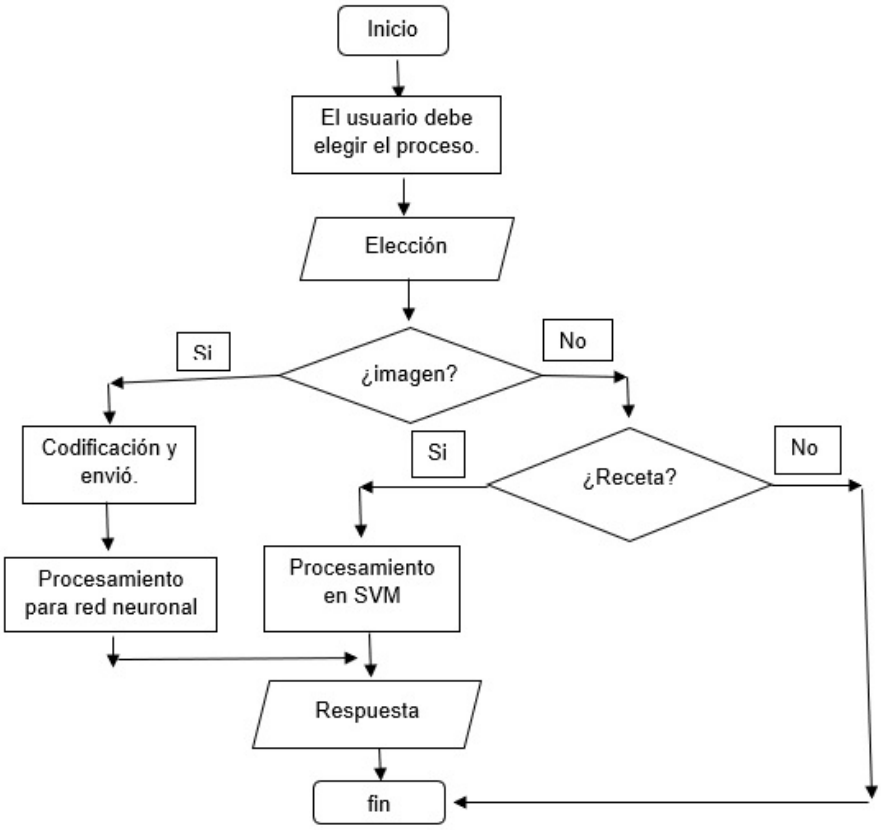


Figura 5. Programación de arquitectura general

Fuente: elaboración propia.

## 2.2 Paso 3: desarrollo de la SVM

La tabla 4 muestra la configuración de la SVM, en el marco de la cual se define un arreglo con las nomenclaturas que corresponden a los desayunos. Los índices de posición corresponden a aquellos que se disponen en el *dataset* que alimenta la SVM y retorna

la predicción realizada. La tabla 5, a su turno, muestra los resultados del entrenamiento de la SVM con variaciones en los porcentajes de entrenamiento y validación.

Tabla 4. Programación configuración máquina de soporte vectorial

```
self.desayunos = [
“PAISA”,
“CAFETERO”,
“ROLO”,
“AMERICANO”,
“FITNESS”,
“TOLIMENSE”,
]
ruta_data =”. /models/data_svc.csv”
data_set = pd.read_csv(ruta_data)
X = data_set.drop(“CLASE”, axis = 1)
Y = dats_set[“CLASE”]
self.svc_classifier = SVC(Kernel = “rbf”)
self.svc_classifier.fit(x_train, y_train)

def predict(self, recipe):
 x = self.svc_classifier.predict(recipe)
 return self.desayunos[int(x)]
```

Fuente: elaboración propia.

Tabla 5. Resultados de entrenamiento con la máquina de soporte vectorial para la predicción de recetas, con variaciones en los porcentajes de entrenamiento y validación

| Porcentaje de entrenamiento | Porcentaje de validación | Resultados |
|---|---|---|
| 20% | 80% | 100% |
| 40% | 60% | 100% |
| 60% | 40% | 100% |
| 80% | 20% | 100% |

Fuente: elaboración propia.

### 2.3 Paso 4: pruebas de entrenamiento

En cuanto al entrenamiento se realizaron diversos procesos que iniciaron con muestras pequeñas (5 categorías, 10 imágenes por categoría), para las cuales se variaron los valores de épocas y tamaño de lote; luego, se iniciaron pruebas con los parámetros utilizados en la prueba de despegue con todo el grupo de datos (17 imágenes, 210 imágenes por categoría); y finalmente se realizaron tres pruebas más con todo el grupo de datos, cambiando los parámetros principales (épocas y tamaño del lote).

Se obtuvieron promedios de precisión del 75,8 % para el reconocimiento de ingredientes con la red neuronal (tabla 6) y de 71, 3 % respecto al de recetas para el entrenamiento realizado con la SVM (tabla 7). En la figura 6 se observa la interfaz terminada: se realizó la predicción de 2 ingredientes (huevos y chocolate) y a partir de ellos se predijo que los ingredientes seleccionados conformarían un desayuno cafetero. Con el fin de obtener la estabilidad del sistema, el proceso fue adelantado 10 veces en grupos de 5, ante lo cual todos los resultados fueron iguales en términos estadísticos (tablas 8 y 9).

Tabla 6. Resultados de cada ingrediente en el entrenamiento con la red neuronal

| Ítem | Descripción | Precisión |
|---|---|---|
| 1 | Huevos | 87 % |
| 2 | Arepas | 91 % |
| 3 | Mantequilla | 79 % |
| 4 | Chocolate | 74 % |
| 5 | Pan | 77 % |
| 6 | Cereales | 79 % |
| 7 | Café | 73 % |
| 8 | Leche | 70 % |
| 9 | Tocino | 73 % |
| 10 | Changua | 74 % |
| 11 | Tamal | 70 % |
| 12 | Papas | 72 % |
| 13 | Calentado | 71 % |
| 14 | Yuca frita | 77 % |
| 15 | Jugo de naranja | 72 % |
| 16 | Yogur | 68 % |
| 17 | Pollo | 83 % |
| Promedio | | 75,8 % |

Fuente: elaboración propia.

Tabla 7. Resultados de cada receta en el entrenamiento con la SVM

| Ítem | Descripción | Precisión |
|---|---|---|
| 1 | Paisa | 69 % |
| 2 | Cafetero | 82 % |
| 3 | Rolo | 72 % |
| 4 | Fitness | 59 % |
| 5 | Tolimense | 71 % |
| 6 | Americano | 75 % |
| Promedio | | 71,3 % |

Fuente: elaboración propia.

Tabla 8. Estabilidad del sistema - ingredientes

| **Técnica** | **Numero de intentos promedio predicción de ingredientes** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Iteración** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| Red neuronal (Café) | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | Ok | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | Ok | NG | OK | OK | OK | OK | OK | OK | OK | OK |
| | NG | NG | NG | NG | NG | NG | NG | NG | NG | NG |
| | OK | OK | OK | NG | OK | OK | OK | OK | OK | OK |
| | NG | OK | NG | NG | NG | NG | OK | NG | NG | NG |
| | NG | NG | NG | NG | NG | NG | NG | NG | NG | NG |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | NG | NG | NG | NG | NG | NG | NG | NG | NG | NG |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Porcentajes | 73,3 % | 73,3 % | 73,3 % | 66,6 % | 73,3 % | 73,3 % | 80 % | 73,3 % | 73,3 % | 73,3 % |

Fuente: elaboración propia.

Tabla 9. Estabilidad del sistema - recetas

| **Técnica** | **Numero de intentos promedio predicción de Recetas** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Iteración** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | NG | NG | NG | NG | NG | NG | NG | NG | NG | NG |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| | NG | NG | NG | NG | NG | NG | NG | NG | NG | NG |
| | OK | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| Porcentajes | 83 % | 83 % | 83 % | 83 % | 83 % | 83 % | 83 % | 83 % | 83 % | 83 % |

Fuente: elaboración propia.


Figura 6. Aplicación para la predicción de ingredientes y recetas

Fuente: elaboración propia.

## 3. DISCUSIÓN

El reconocimiento de imágenes se puede aplicar para predecir tareas domésticas. En esta investigación se usó para predecir los ingredientes y las recetas; sin embargo, se encontró que esta metodología se ha utilizado en otras áreas: la asistencia a personas con alguna discapacidad visual, mediante el reconocimiento de la moneda local [27]; las ventajas y aplicaciones de la domótica para ayudar personas con discapacidad [28]; y el reconocimiento de las actividades por medio de una cámara web que realiza una persona en la cocina [29]. A medida que avanza la tecnología se busca que estos procesos se realicen con procesadores de bajo costo como Raspberry [30], usado con anterioridad para supervisar y controlar procesos en el ámbito industrial [18], o el Node MCU ESP82866, empleado para predecir la escena en la cocina por medio de sensores IoT y que se asocia a un servidor IoT, dado que este hace posible la gestión de entornos domésticos [19] [31].

A la par con lo dicho, se han desarrollado investigaciones sobre robots de servicio inteligentes que operen en entornos humanos estándar y automaticen tareas comunes [32]. Cabe señalar que la navegación autónoma de robots es uno de los principales problemas que plantea su desarrollo debido a su complejidad y dinamismo [33]; por ello, se espera integrar estos sistemas de reconocimiento de alimentos al robot para que este tenga un aprendizaje autónomo y pueda desempeñar mejor sus funciones.

Los resultados obtenidos con esta investigación son similares similar a los obtenidos por Leal, Ochoa, y García [8], quienes usaron la técnica de SVM para identificar fracturas naturales en pozos de yacimientos de hidrocarburos con una exactitud que osciló entre 72,3 % y 82,2 % [8]; ello demuestra la gran flexibilidad que reviste el modelo.

La técnica de SVM también se ha empleado en sistemas eléctricos, enfocándose en temas como predicción del mercado eléctrico, pérdidas no técnicas de electricidad (hurto), energías alternativas y transformadores, entre otros [34]. Además, también se ha hecho uso de SVM en iniciativas de tipo social, como la identificación del grado de riesgo psicosocial en docentes de colegios públicos en Colombia [35].

En cuanto al uso de TensorFlow, se observó que este admite una variedad de aplicaciones con un enfoque en capacitación e inferencia en redes neuronales profundas [36]; y que se están desarrollando herramientas paralelas como TensorD, la cual proporciona métodos de descomposición del tensor, así como operaciones básicas del tensor para facilitar la práctica de los métodos tensoriales en visión artificial, aprendizaje profundo y otros [37]. Así entonces, se espera usar la herramienta compuesta en esta investigación para aumentar el número de ingredientes reconocidos (100) y recetas conocidas (200), e integrarla a futuro con otro sistema para el surtimiento automático de alimentos o con diversas aplicaciones en la industria alimentaria, habida cuenta de que la evolución tecnológica permite disponer de soluciones altamente flexibles, de fácil acceso y con valores de inversión más bajo [38].

## 4. CONCLUSIONES

Los datos obtenidos permiten inferir que el sistema concebido funciona y puede implementarse en entornos reales. Se determinó igualmente que el uso de SVM junto con TensorFlow representa una gran herramienta de clasificaciones. Se aspira en este sentido a extender este modelo en una segunda fase con el uso de microprocesadores de bajo consumo como Raspberry Pi y la cámara Pi, en aras de realizar este reconocimiento en un entorno real y aumentar tanto la cantidad de ingredientes reconocidos (100) como el reconocimiento en recetas (20), a fin de que en el futuro se pueda implementar esta herramienta con un sistema para el sistema de surtimiento automático de alimentos. Solo resta apuntar que, en cuanto al análisis realizado para verificar la estabilidad, se observó que los resultados fueron iguales en términos estadísticos.

## AGRADECIMIENTOS

Se agradece por su colaboración a las universidades Nacional de Colombia (sede Manizales) y de Caldas, a Mabe y a todos los integrantes del proyecto "Prototipo computacional para la fusión y análisis de grandes volúmenes de datos en entornos IoT (internet de las cosas) a partir de técnicas de *machine learning* y arquitecturas seguras entre sensores para caracterizar el comportamiento e interacción de los usuarios en un ecosistema de *connected home"* (código 36715).

## REFERENCIAS

[1] A. Krizhevsky *et al.,* "ImageNet Classification with Deep Convolutional Neural Networks", *NIPS,* vol. 1, pp. 1097-1105, 2012. DOI: 10.1201/9781420010749

[2] G. P. García, *Reconocimiento de imágenes utilizando redes neuronales artificiales,* Proyecto fin de Máster, Universidad Complutense de Madrid, Madrid, 2013.

[3] Y. LeCun *et al.,* "Deep learning", *Nature Methods,* vol. 521, n.°1, pp. 436-444, 2015. DOI: 10.1038/nature14539

[4] F. S. Pardo, "Aplicación del modelo Bag-of-Words al reconocimiento de imágenes", proyecto de fin de carrera, Universidad Carlos III de Madrid, Madrid, 2009.

[5] M. L Guevara et al., "Faces Detection in Digital Images Using Cascade Classifiers", *Scientia et Technica,* vol. 38, n.° 38, pp. 1–6, 2008.

[6] G. A. Betancourt "Las máquinas de soporte vectorial (SVMs)", *Scientia Et Technica,* vol. 27, pp. 67-72, 2005.

[7] J. E. Hurtado *et al.,* "Clasificación de Señales Sísmicas por Medio de Onditas y Máquinas de Soporte Vectorial", en *Primer simposio colombiano de sismología*, Manizales, Colombia, 2002.

[8] J. A. Leal *et al.,* "Identification of natural fractures using resistive image logs, fractal dimension and support vector machines", *Ingeniería e Investigación*, vol. 36, n.° 3, pp. 125-132, 2016. DOI: 10.15446/ing.investig.v36n3.56198.

[9] I. C. Guzmán et al., "Wavelet denoising of partial discharge signals and their pattern classification using artificial neurol networks and support vector machines", *DYNA,* vol. 84, n.° 203, pp. 240-248, 2017. DOI: 10.15446/dyna.v84n203.63745.

[10] J. E. Espinosa *et al.,* "Kernel Methods for improving Text Search Engines Transductive Inference by Using Support Vector Machines", *TECCIENCIA*, vol. 12, n.° 52, pp. 51-60, 2017. DOI: 10.18180/tecciencia.2017.22.6.

[11] J. R. Wilches *et al.,* "A VoIP classifier for carrier grade based on Support Vector Machines", *DYNA,* vol. 84, n.° 202, pp. 75-83, 2017. DOI: 10.15446/dyna.v84n202.60975.

[12] R. Cabañas *et al.,* "InferPy: Probabilistic modeling with TensorFlow made easy", *Knowledge-Based Systems*, vol. 168, pp. 25-27, 2019. DOI: 10.1016/j.knosys.2018.12.030.

[13] M. Liu y D. Grana, "Accelerating geostatistical seismic inversion using TensorFlow: A heterogeneous distributed deep learning framework", *Computers and Geosciences,* vol. 124, pp. 37-45, 2019. DOI: 10.1016/j.cageo.2018.12.007.

[14] J. R. Vázquez et al., "Fusing TensorFlow with building energy simulation for intelligent energy management in smart cities", *Sustainable Cities and Society*, vol. 45, pp. 243-257, 2019. DOI: 10.1016/j.scs.2018.11.021.

[15] S. Liu *et al.,* "The research of virtual face based of deep convolutional generative adversarial networks using TensorFlow", *Physica A: Statistical Mechanics and its applications,* vol. 521, pp. 667-580, 2019.

[16] X. Grandio (2017, 14 de julio) "Blog: Que es TensorFlow: Aplicaciones del sistema de inteligencia artificial de Google". [internet]. Disponible en https://marketing4ecommerce.net.

[17] P. Parsch y A. Masrur, "On Reliable Communication in Transmit-only Networks for Home Automation", *Journal of Network and Computer Applications,* vol. 101, pp. 96-110, 2017. DOI: 10.1016/j.jnca.2017.10.023.

[18] S. A. Castro *et al.,* "Supervisión y Control Industrial a través de Teléfonos Inteligentes usando un computador de placa única Raspberry Pi", *Inf. Tecnol,* vol. 27, n.° 2, 2016. DOI: 10.4067/S0718-07642016000200015.

[19] J. A. Asensio *et al.,* "Emulating home automation installations through component-based web technology", *Future Generation Computer Systems,* vol. 93, pp. 777-791, 2017. DOI: 10.1016/j.future.2017.09.062.

[20] J. M. Marín (2014). "Introducción a las redes neuronales aplicadas". [internet]. Disponible en http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/DM/tema3dm.pdf.

[21] A. Serrano, E. Soria y J. D. Martin (2009). "Redes Neuronales Artificiales". [internet]. Disponible en http://ocw.uv.es/ingenieria-y-arquitectura/1-2/libro_ocw_libro_de_redes.pdf.

[22] D. Calvo (2018). "Función de coste – Redes Neuronales". [internet]. Disponible en http://www.diegocalvo.es/funcion-de-coste-redes-neuronales/.

[23] J. F. Quesada (1997). "Características de las redes neuronales". [internet]. Disponible en https://thales.cica.es/rd/Recursos/rd98/TecInfo/07/capitulo3.html.

[24] E. Carmona (2014). "Tutorial sobre Máquinas de Vectores Soporte (SVM)" [internet]. Disponible en http://www.ia.uned.es/~ejcarmona/publicaciones/[2013-Carmona]%20SVM.pdf.

[25] J. A. Reséndiz (2006). "Las máquinas de vectores de soporte para identificación en línea". [internet]. Disponible en https://www.ctrl.cinvestav.mx/~yuw/pdf/MaTesJAR.pdf.

[26] J. Echeverri *et al*., "Mejoramiento de imágenes usando funciones de base radial", *Revista ingenierías universidad de Medellín,* vol. 8, n.° 15, sup., 1, pp. 21-28, 2009.

[27] B. C. Gayón, *Desarrollo de una Aplicación para Reconocimiento de Billetes por medio de Procesamiento de Imágenes con Diversidad Visual Basada en Tecnología Android*, proyecto de fin de carrera, Universidad Libre, Bogotá, 2017.

[28] F. J. Rodríguez, *Automatización Mediante Equipos EIB de una Cocina Adaptada Dentro del Entorno del Robot Asistencial ASIBOT*, proyecto de fin de carrera, Universidad Carlos III, Madrid, 2010.

[29] F. J. García, *Reconocimiento de objetos en una cocina con una webcam*, proyecto de fin de carrera, Universidad Carlos III, Madrid, 2009.

[30] R. Colombo, *"Deep Learning" para reconocimiento de imágenes en Raspberry Pi 2,* proyecto de fin de carrera, Universidad de la Laguna, Santa Cruz de Tenerife, 2016.

[31] Y. Muñoz *et al.,* "Análisis de la escena en la cocina por medio de sensores IoT Diseñados basados en el microcontrolador node MCu ESP8266 y conectados al servidor ThingSpeak", *Inf Tecnol,* vol. 30, pp. 173-170, 2019. DOI: 10.4067/S0718-07642019000500173.

[32] R. B. Rusu *et al.,* "Robots in the kitchen: Exploiting ubiquitous sensing and actuation", *Robotics and Autonomous Systems,* vol. 56, n.° 10, pp. 844-856, 2008. DOI: 10.1016/j.robot.2008.06.010.

[33] Y. Quiñonez *et al.,* "Aplicación de técnicas evolutivas y visión por computadora para navegación autónoma de robots utilizando un TurtleBot", *RISTI,* vol. 3, pp. 93-105, 2015. DOI: 10.17013/risti.e3.93-105.

[34] J. Estupiñán *et al.,* "Implementación de algoritmos basados en máquinas de soporte vectorial (SVM) para sistemas eléctricos: revisión de tema", *Tecnura*, vol. 20, n.° 10, pp. 149-170, 2016. DOI: 10.14483/udistrital.jour.tecnura.2016.2.a11.

[35] R. Mosquera et al., "Máquinas de Soporte Vectorial, Clasificador Naïve Bayes y Algoritmos Genéticos para la Predicción de Riesgos Psicosociales en Docentes de Colegios Públicos Colombianos", *Inf. Tecnol,* vol. 29, n.° 6, pp. 153-162, 2018. DOI: 10.4067/s0718-07642018000600153.

[36] M. Abadi *et al.,* "TensorFlow: A system for large-scale machine learning. TensorFlow: A System for Large-Scale Machine Learning", en *XII Simposio USENIX sobre diseño e implementacion de sistemas operativos OSDI'16,* Savannah, 2016.

[37] L. Hao *et al.,* "TensorD: A tensor decomposition library in TensorFlow", *Neurocomputing,* vol. 318, pp. 196-200, 2018. DOI: 10.1016/j.neucom.2018.08.055.

[38] V. M. Araújo y M. P. Cota, "Software como um Serviço: Uma visão holística", *RISTI*, n.° 19, pp. 145-157, 2016. DOI: 10.17013/risti.19.145-157.