

Universidad Rafael Landívar  
Campus central Quetzaltenango  
Facultad de ingeniería  
Ingeniería en informática y Sistemas  
Docente: Ing. Dhaby Xiloj  
Inteligencia Artificial

**Proyecto Final de Curso**

Reconocimiento de madurez de mangos por medio de una RNA

Wellington José Eduardo Cuá Gutiérrez	1511612
Pedro Luis Velásquez Moreno	1505513

Quetzaltenango, 01 de mayo de 2019

# DETECCION DE PATRONES POR MEDIO DE REDES NEURONALES

## MANGOS

### **Producción de Mango en Guatemala**

El mango está reconocido en la actualidad como uno de los 3 ó 4 frutos tropicales más finos en Guatemala. Ha estado bajo cultivo desde los tiempos prehistóricos

El mango está reconocido en la actualidad como uno de los 3 ó 4 frutos tropicales más finos en Guatemala. Ha estado bajo cultivo desde los tiempos prehistóricos. El árbol de mango ha sido objetivo de gran veneración en la India y sus frutos constituyen un artículo estimado como comestibles a través de los tiempos.

Fruto carnoso, sabroso y refrescante, conocido también como “melocotón de los trópicos”. Miembro más importante de la familia de las Anacardiáceas o familia de marañón.

Florece y fructifica de manera muy semejante al aguacate, es decir, en grandes panículas muy ramificadas que aparece en las extremidades de ramas del año que posee suficiente madurez.

Con respecto a la formación del árbol si es necesario invertir con la poda, muy particularmente en la selección de las ramas principales que iniciaran la copa. Si bien es cierto que los arboles de esta pueden formar su estructura normal sin ninguna ayuda de la poda, también es verdad que el mango, en gran número de variedades, tiende con frecuencia a emitir cuando joven brotes muy verticales, con ángulos de inserción muy cerrados.

En la época de producción de mango en Guatemala es de febrero a junio, y el 80% se destina hacia Estados Unidos y el 20% restante se comercializa en el mercado de Centroamérica y Europa.

Actualmente se cultiva en todo el mundo especialmente en zonas trópicas y subtropicales. La planta es considerada “siempre verde” su desarrollo varía dependiendo del origen de la planta.

El mango constituye un árbol de tamaño, de 10-30 metros de altura, con una Copa densa y ampliamente oval o globular. Las ramas son gruesas y robustas, las hojas son alternas, espiadas irregularmente a lo largo de las ramas.

Debemos tomar en cuenta los análisis de suelo y foliar muy importante para elaborar el programa de fertilización.

Cuando se carece de muestreos se debe incorporar suficiente materia orgánica a la tierra, con la cual se llenaran las bolsas de polietileno; además efectuar las aplicaciones de fertilizante completos. Es importante suministrar elementos menores a plantas jóvenes por medio de aspersiones, para corregir eficiencia de dichos elementos especialmente Zinc.

Los mangos pueden producirse a partir de las semillas desde el verano al otoño. La semilla debe retirarse de la cascara y plantarse sobre el terreno con la parte más protuberante hacia arriba. También se sabe que la capacidad de germinación de la semilla del mango puede desaparecer al cabo de 8 días de haber recogido el fruto. De todas maneras se puede realizar la prueba, plantando varias semillas procedentes de diferentes frutos comprados en diferentes lugares.

El mango se adapta bien a climas tropicales o sub-tropicales secos cuyos rangos de temperatura óptima media se encuentren entre los 20 y 25°C, teniendo como mínimo temperaturas mayores a 15°C, ya que no soporta heladas. La humedad relativa debe situarse por debajo de 70%. Este frutal se adapta a cualquier tipo de suelos que sea bien drenados, pero se adapta mejor a suelos profundos (de 1.5 a 2 metros) de textura intermedia (franca arcillosa, franca limosa o franca arenosa), con un pH que varíe entre 5.5 a 7.5.

## Aspectos Productivos del mango

### Épocas de siembra

El mango como todo producto tiene temporadas en las cuales se da la siembra, en la temperatura de invierno tiene mayor importancia en el Altiplano Occidental, el mango de suelo es recomendable sembrarlo a partir del 12 de febrero al 28 de junio ya que si las siembras se realizan a una fecha muy temprana pueden sufrir las lluvias de invierno de igual manera si estas siembras se llegaran a tardar pueden estar expuestas a temperaturas muy altas o bajas durante la segunda quincena de noviembre provocando fallos en la siembra.



### Principales departamentos productores de mango en Guatemala.

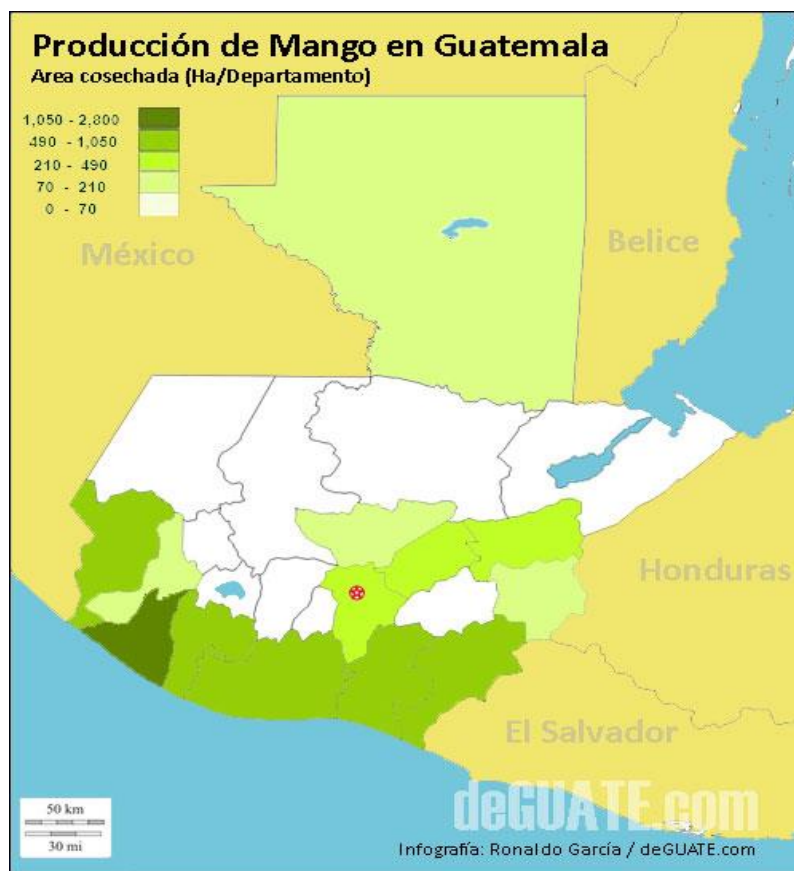
La producción nacional de mango se encuentra distribuida de la siguiente forma: Retalhuleu (28%), Santa Rosa (13%), Suchitepéquez (10%), Escuintla (10%), San Marcos (7%), Jutiapa

(7%), El Progreso (6%) y los demás departamentos de la República suman el (18%) restante.



### Área cosechada

El 62% de la superficie cosechada se encuentra concentrada en 4 departamentos: Retalhuleu (33%), Santa Rosa (12%), Suchitepéquez (10%) y Jalapa (7%).



## Producción y Rendimiento (2008-2013)

En la siguiente tabla podemos apreciar el área cosechada de mango (en manzana), la producción de mango (en quintales) y el rendimiento (en quintales por manzana):

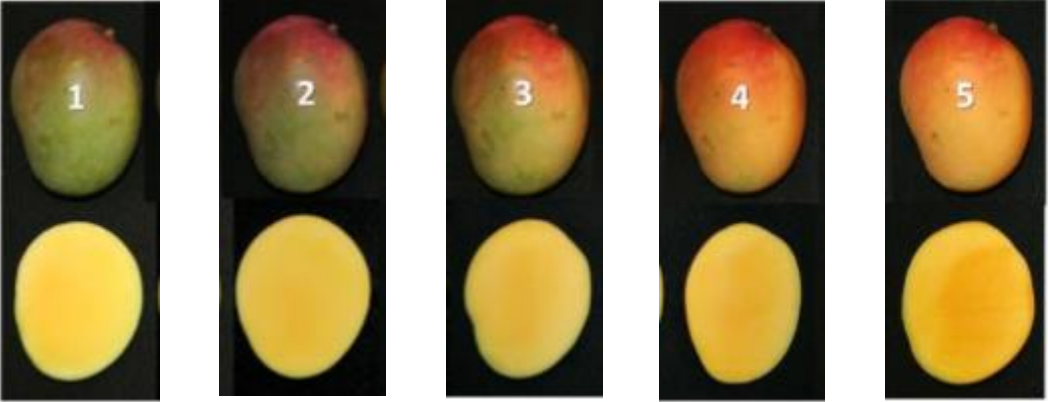
Año calendario	Área cosechada (manzanas)	Producción (quintales)	Rendimiento (qq/mz)
2008	12,750.00	2,491,703.50	195.43
2009	13,000.00	2,569,239.80	197.63
2010	13,048.50	2,425,497.90	185.88
2011	13,800.00	2,563,400.00	189.00
2012 p/	13,400.00	2,502,100.00	186.30
2013 e/	13,500.00	2,526,000.00	187.00

## Costos de Producción (2012-2013)

### RAZONES POR LAS CUALES SE ELIGIÓ MANGO:

- Se cuentan con siembras en áreas cercanas a Quetzaltenango, lo cual facilita las muestras y obtención de información directa de las personas implicadas en su siembra, cosecha y venta.
- Debido a que dicha fruta se encuentra en temporada, el consumo de este producto incrementa, por esta razón, es fácil recabar información de los consumidores en como eligen el mango.
- Debido a que su precio es bajo, se compraron diversas unidades para la toma de muestras y así entrenar la red neuronal.

## FACTORES TÉCNICOS EN LA CLASIFICACIÓN DEL MANGO:

Etapa	Etapa 1	Etapa 2	Etapa 3	Etapa 4	Etapa 5
<b>Descripción</b>	Color verde claro en la cosecha, listo para ser enviado.	El mango comienza a madurar pasa de un verde más claro a amarillo. En esta etapa el mango es firme y apropiado para el envío al comercio minorista.	Representa un mango medio-verde y medio-amarillo pero todavía es firme y se acerca a la madurez.	Muestra claramente el proceso de maduración como el mango se vuelve amarillo naranja y comienza a suavizar. Los aromas fragantes se intensificarán desde el extremo del tallo. Ahora la fruta está lista para ser exhibida en las tiendas al por menor.	Se muestra en el mango un color naranja dorado y es suave al tacto. Una característica normal es la presencia de arrugas. Esta etapa es altamente recomendable comer a tiempo para el mejor gusto.
<b>Imagen</b>					

## RED NEURONAL

Las redes neuronales (también conocidas como sistemas conexionistas) son un modelo computacional basado en un gran conjunto de unidades neuronales simples (neuronas artificiales), de forma aproximadamente análoga al comportamiento observado en los axones de las neuronas en los cerebros biológicos. Cada unidad neuronal está conectada con muchas otras y los enlaces entre ellas pueden incrementar o inhibir el estado de activación de las neuronas adyacentes. Cada unidad neuronal, de forma individual, opera empleando funciones de suma. Puede existir una función limitadora o umbral en cada conexión y en la propia unidad, de tal modo que la señal debe sobrepasar un límite antes de propagarse a otra neurona. Estos sistemas aprenden y se forman a sí mismos, en lugar de ser programados de forma explícita, y sobresalen en áreas donde la detección de situaciones o características es difícil de expresar con la programación convencional.

Las redes neuronales suelen consistir en varias capas o un diseño de cubo, y la ruta de la señal atraviesa de adelante hacia atrás. Propagación hacia atrás es donde se utiliza la estimulación hacia adelante o en el "frente" para restablecer los pesos de las unidades neuronales y esto a veces se realiza en combinación con una formación en la que se conoce el resultado correcto. Las redes modernas son un poco más libres en el sentido de que fluye en términos de estimulación e inhibición con conexiones que interactúan de una manera mucho más caótica y compleja. Redes neuronales dinámicas son lo más avanzadas en que pueden dinámicamente, formar nuevas conexiones e incluso nuevas unidades neuronales.

El objetivo de la red neuronal es resolver los problemas de la misma manera que el cerebro humano, aunque varias redes neuronales son más abstractas. Proyectos de redes neurales modernas suelen trabajar con unos pocos miles a unos pocos millones de unidades neuronales y millones de conexiones, que sigue siendo varios órdenes de magnitud menos complejo que el cerebro humano y más cercano a la potencia de cálculo de un gusano.

Las redes neuronales se basan en los números reales, con el valor del núcleo y del axón siendo típicamente una representación entre 0,0 y 1.

Un aspecto interesante de estos sistemas es que son impredecibles en su éxito con el autoaprendizaje. Después del entrenamiento, algunos se convierten en grandes solucionadores de problemas y otros no funcionan tan bien. Con el fin de capacitar a ellos, varios miles de ciclos de interacción que se producen normalmente.

Las redes neuronales se han utilizado para resolver una amplia variedad de tareas, como la visión por computador y el reconocimiento de voz, que son difíciles de resolver usando ordinaria de programación basado en reglas.

## Back propagation

La propagación hacia atrás de errores o retropropagación (del inglés backpropagation) es un algoritmo de aprendizaje supervisado que se usa para entrenar redes neuronales artificiales. El algoritmo emplea un ciclo propagación – adaptación de dos fases. Una vez que se ha aplicado un patrón a la entrada de la red como estímulo, este se propaga desde la primera capa a través de las capas siguientes de la red, hasta generar una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas.

Las salidas de error se propagan hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa oculta que contribuyen directamente a la salida. Sin embargo las neuronas de la capa oculta sólo reciben una fracción de la señal total del error, basándose aproximadamente en la contribución relativa que haya aportado cada neurona a la salida original. Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido una señal de error que describa su contribución relativa al error total.

La importancia de este proceso consiste en que, a medida que se entrena la red, las neuronas de las capas intermedias se organizan a sí mismas de tal modo que las distintas neuronas aprenden a reconocer distintas características del espacio total de entrada. Después del entrenamiento, cuando se les presente un patrón arbitrario de entrada que contenga ruido o que esté incompleto, las neuronas de la capa oculta de la red responderán con una salida activa si la nueva entrada contiene un patrón que se asemeje a aquella característica que las neuronas individuales hayan aprendido a reconocer durante su entrenamiento.

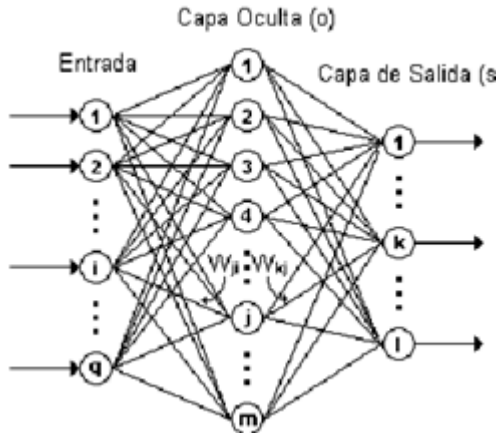
## ¿Cómo funciona el algoritmo?

La red retro propagada trabaja bajo aprendizaje supervisado y por tanto necesita un conjunto de entrenamiento que le describa cada salida y su valor de salida esperado de la siguiente forma:

$$(p_1, t_1), (p_2, t_2), \dots, (p_q, t_q) \quad (1)$$

Donde  $p_q$  es una entrada a la red y  $t_q$  es la correspondiente salida deseada para el patrón  $q$ -ésimo. El algoritmo debe ajustar los parámetros de la red para minimizar el error cuadrático medio. Es importante recalcar que no existe una técnica para determinar el número de capas ocultas, ni el número de neuronas que debe contener cada una de ellas para un problema específico, esta elección es determinada por la experiencia del diseñador, el cual debe cumplir con las limitaciones de tipo computacional. Cada patrón de entrenamiento se propaga a través de la red y sus parámetros para producir una respuesta en la capa de salida, la cual se compara con los patrones objetivo o salidas deseadas para calcular el error en el aprendizaje, este error marca el camino más adecuado para la actualización de los pesos y ganancias que al final del entrenamiento producirán una respuesta satisfactoria a todos los patrones de entrenamiento, esto se logra minimizando el error cuadrático medio en cada iteración del proceso de aprendizaje.





- q: Número de componentes del vector de entrada.
- m: Número de neuronas de la capa oculta.
- l: Número de neuronas de la capa de salida.

Para iniciar el entrenamiento se le presenta a la red un patrón de entrenamiento, el cual tiene q componentes como se describe en la ecuación:

$$p = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_i \\ \vdots \\ p_q \end{bmatrix} \quad (2)$$

Cuando se le presenta a la red un patrón de entrenamiento, este se propaga a través de las conexiones existentes produciendo una entrada neta n en cada una de las neuronas de la siguiente capa, la entrada neta a la neurona j de la siguiente capa debido a la presencia de un patrón de entrenamiento, o en la entrada está dada por la ecuación 3, nótese que la entrada neta es el valor justo antes de pasar por la función de transferencia:

$$n_j^0 = \sum_{i=1}^q (w_{ji}^0 * p_i) + b \quad (3)$$

Ecuación 3

- $w_{ji}$ : Peso que une la componente i de la entrada con la neurona j de la capa oculta.

- $P_i$ : Componente  $i$  del vector  $p$  que contiene el patrón de entrenamiento de  $q$  componentes.
- $b_j$ : Ganancia de la neurona  $j$  de la capa oculta.

Donde (0) representa la capa a la que pertenece cada parámetro, es este caso la capa oculta. Cada una de las neuronas de la capa oculta tiene como salida  $a_j$  dada por:

$$a_j^0 = f^0\left(\sum_{i=1}^q (w_{ji}^0 * p_i) + b_j^0\right) \quad (4)$$

$f^0$ : Función de transferencia de las neuronas de la capa oculta. Las salidas  $a_j^0$  de las neuronas de la capa oculta (de  $m$  componentes) son las entradas a los pesos de conexión de la capa de salida, este comportamiento está descrito por:

$$n_k^s = \sum_{j=1}^m (w_{kj}^s * a_j^0) + b_k^s \quad (5)$$

- $w_{kj}$ : Peso que une la neurona  $j$  de la capa oculta con la neurona  $k$  de la capa de salida, la cual cuenta con  $s$  neuronas.
- $a_j$ : Salida de la neurona  $j$  de la capa oculta, la cual cuenta con  $m$  neuronas.
- $b_k$ : Ganancia de la neurona  $k$  de la capa de salida.
- $n_k$ : Entrada neta a la neurona  $k$  de la capa de salida.

La red produce una salida final:  $a_k^s = f^s(n_k^s) \quad (6)$

$f^s$ : Función de transferencia de las neuronas de la capa de salida.

La salida de la red de cada neurona  $a_k$  se compara con la salida deseada  $t_k$  para calcular el error en cada unidad  $\delta_k = (t_k - a_k^s)$ . El error debido a cada patrón  $p$  propagado está dado por:

$$ep^2 = 1/2 \sum_{k=1}^l (\delta_k^2)$$

- $ep^2$ : Error cuadrático medio para cada patrón de entrada  $p$ .

- o2: Error en la neurona k de la capa de salida con l neuronas.

Este proceso se repite para el número total de patrones de entrenamiento (r), para un proceso de aprendizaje exitoso el objetivo del algoritmo es actualizar todos los pesos y ganancias de la red minimizando el error cuadrático medio total descrito en:

$$e^2 = \sum_{p=1}^r (ep^2)$$

e2:: Error total en el proceso de aprendizaje en una iteración después de haber presentado a la red los r patrones de entrenamiento. El error que genera una red neuronal en función de sus pesos, genera un espacio de n dimensiones, donde n es el número de pesos de conexión de la red, al evaluar el gradiente del error en un punto de esta superficie se obtendrá la dirección en la cual la función del error tendrá un mayor crecimiento, como el objetivo del proceso de aprendizaje es minimizar el error, debe tomarse la dirección negativa del gradiente para obtener el mayor decremento del error y de esta forma su minimización, condición requerida para realizar la actualización de la matriz de pesos en el algoritmo Backpropagation.

# APLICACIÓN DE RECONOCIMIENTO

## Descripción

Se debe crear una red neuronal artificial, capaz de poder analizar una imagen en la cual se encuentre un mango, y poder a partir de esta saber un aproximado del nivel de madurez del mango.

Para poder realizar este análisis correctamente, se tiene que tomar en cuenta que de la fotografía lo único que necesitamos es una región del mango que nos muestre la piel de dicha fruta. Para ello es necesario hacerle un tratamiento a la imagen para poder encontrar esa región, los pasos necesarios se describen a continuación:

- Crear máscaras de color: en este proceso se hace una copia exacta de la imagen para luego pasarle un filtro de tal manera que solo los colores que se establecen se marquen, en este caso se busca un amarillo, por lo cual se necesitan crear dos máscaras pues los amarillos se encuentran en los extremos diferentes del espectro visible. cuando se le pasa este filtro a la copia todo lo que no sea amarillo queda negro y los tonos de amarillos buscados quedan en blanco.
- Buscar el área del mango: con la máscara tenemos el área del mango en blanco, lo que nos facilita dibujar un elipse que encierre la parte blanca, usando los mínimos y el centro de la elipse se encuentran los puntos para recortar el área de la piel que necesitamos.

Ahora ya tenemos el recorte de la piel de un mango. Para poder entrenar la neurona necesitamos muchos ejemplos por cada estado de madurez del tomate, es necesario tener una buena base de datos para cada estado de madurez.

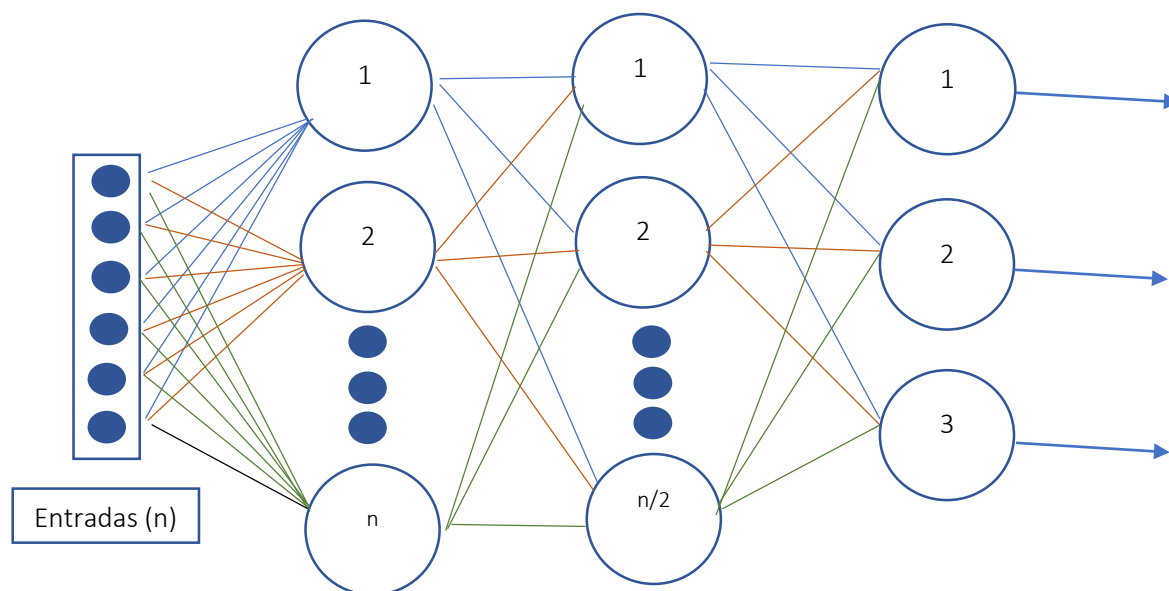
Para poder entrar lo que se necesita son los píxeles de cada recorte, se tiene que tomar en cuenta que hasta ahora tenemos un recorte de diferente tamaño para cada ejemplo de mango, por lo cual hay que redimensionar al tamaño deseado con la pérdida mínima, y después sacarle los píxeles, en cada pixel de un RGB hay tres datos correspondientes a cada color.

Todos estos datos conforman una sola fila en nuestra matriz de entrada a esta fila tenemos que agregarle las salidas deseadas en este caso:

- Podridos: 1 0 0
- Maduros: 0 1 0
- Verdes: 0 0 1

el resultado de este proceso es un archivo .csv que contiene los pixeles de cada una de las imágenes más su salida deseada por cada fila. la cantidad de filas estará dado por la cantidad de imágenes que tengamos para entrenar y la cantidad de columnas será la multiplicación de las dimensiones de las imágenes por tres datos por pixel.

### RED NEURONAL A UTILIZAR



- El número de neuronas para la primera capa oculta es igual al número de neuronas en la capa de entrada.
- El número de neuronas en la segunda capa es  $n/2$
- 3 neuronas en la capa de salida

### HERRAMIENTAS A UTILIZAR

#### Neurolab

NeuroLab - una biblioteca de redes neuronales algoritmos básicos con configuraciones de red flexibles y algoritmos de aprendizaje para Python. Para simplificar el uso de la biblioteca, la interfaz es similar al paquete de red neuronal de la caja de herramientas (NNT) de MATLAB (c). La biblioteca se basa en la numpy paquete ( <http://numpy.scipy.org> scipy.optimize), se utilizan unos algoritmos de aprendizaje ( <http://scipy.org> ).

#### OpenCV

OpenCV (Open Source Computer Vision Library) es una fuente abierta de visión por ordenador y la biblioteca de software de aprendizaje de máquina. OpenCV fue construido














para proporcionar una infraestructura común para aplicaciones de visión artificial y acelerar el uso de la percepción de la máquina en los productos comerciales. Al ser un producto de licencia BSD, OpenCV hace que sea fácil para las empresas a utilizar y modificar el código.

La biblioteca cuenta con más de 2500 algoritmos optimizados, que incluye un amplio conjunto de clásico y de visión artificial y algoritmos de aprendizaje automático con tecnología de última generación. Estos algoritmos se pueden utilizar para detectar y reconocer las caras, identificar objetos, clasificar las acciones humanas en los vídeos, los movimientos de cámara pista, moviendo el seguimiento de objetos, extraer modelos 3D de objetos, producen nubes de puntos 3D a partir de cámaras estéreo, fusionar imágenes juntos para producir una alta resolución la imagen de una escena entera, encontrar imágenes similares de una base de datos de imágenes, eliminar los ojos rojos de las imágenes tomadas con flash, seguir los movimientos de los ojos, reconocer el paisaje y establecer marcadores para revestirlo de realidad aumentada, etc. OpenCV tiene más de 47 mil personas de usuario la comunidad y la estimación del número de descargas que superan los 14 millones . La biblioteca se utiliza ampliamente en las empresas, grupos de investigación y por los organismos gubernamentales.

Tiene interfaces de C ++, C, Python, Java y MATLAB y es compatible con Windows, Linux, Android y Mac OS. OpenCV se inclina principalmente hacia las aplicaciones de visión en tiempo real y se aprovecha de instrucciones MMX y SSE cuando esté disponible. A todas las funciones de CUDA y OpenCL interfaces se están desarrollando de forma activa en este momento. Hay más de 500 algoritmos y cerca de 10 veces el número de funciones que lo componen o apoyan esos algoritmos. OpenCV está escrito de forma nativa en C ++ y tiene una interfaz con plantilla que funciona perfectamente con contenedores STL.

## CARPETAS Y ARCHIVOS

A continuación se detalla el nombre de las carpetas necesarias para el correcto funcionamiento de la RNA, así como los archivos necesarios para entrenar otra RNA, con archivos diferentes.

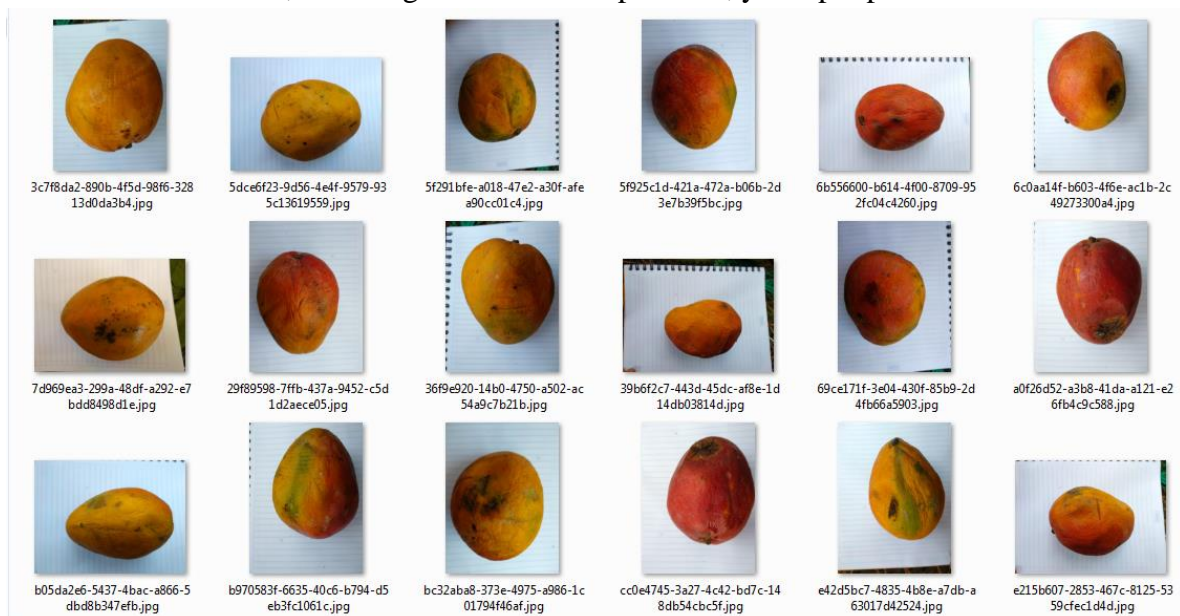
	Fotos de Prueba	26/04/2019 12:10 a...	Carpeta de archivos	
	Interfaz	26/04/2019 10:21 a...	Carpeta de archivos	
	MangosBuenos	26/04/2019 01:51 a...	Carpeta de archivos	
	MangosMalos	26/04/2019 12:10 a...	Carpeta de archivos	
	MangosVerdes	26/04/2019 01:52 a...	Carpeta de archivos	
	RecortesMangosBuenos	26/04/2019 01:55 a...	Carpeta de archivos	
	RecortesMangosMalos	26/04/2019 01:55 a...	Carpeta de archivos	
	RecortesMangosVerdes	26/04/2019 01:55 a...	Carpeta de archivos	
	DatosEntradaRNA.py	25/04/2019 10:00 ...	Archivo PY	3 KB
	RecorteMango.py	25/04/2019 11:20 ...	Archivo PY	4 KB
	RNA.py	26/04/2019 02:41 a...	Archivo PY	2 KB
	Trained-RNA.tmt	26/04/2019 10:14 a...	Archivo TMT	52,173 KB
	TrainingData.csv	26/04/2019 01:55 a...	Microsoft Excel C...	421 KB

Se detallan los archivos y carpetas necesarias para el funcionamiento adecuado, de igual forma el orden en el que deben ser ejecutados, los archivos cuya explicación se omite, son archivos, que se generan tras ejecutar un programa o no son indispensables para el funcionamiento:

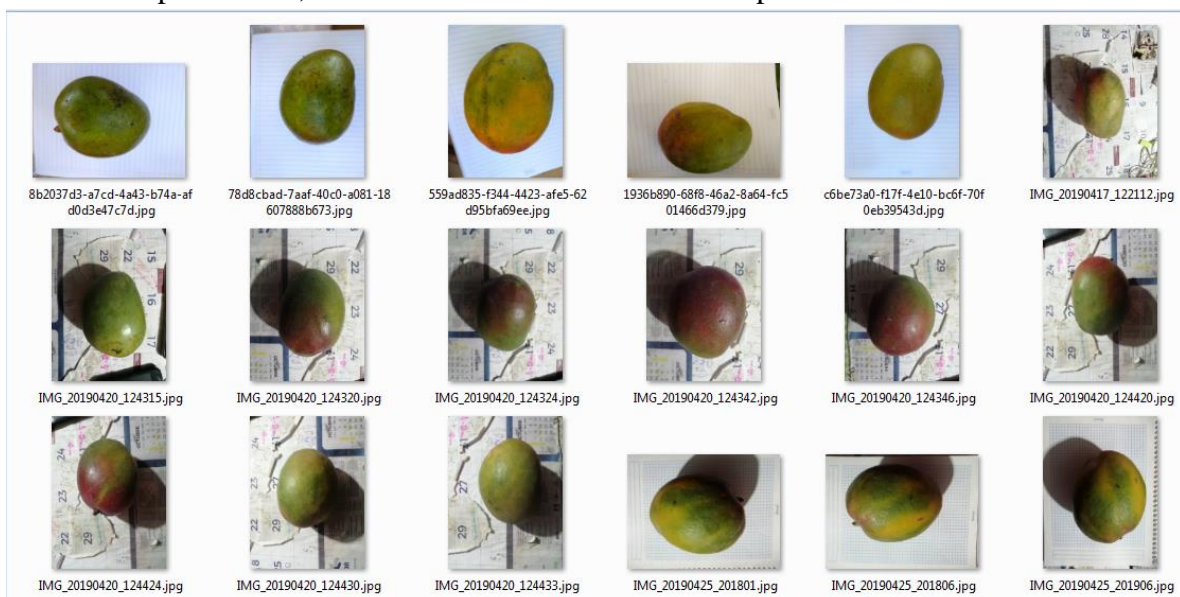
1. **MangosBuenos:** se debe colocar las imágenes con mangos buenos considerando el color, forma, tamaño, en esta clasificación solo entran mangos con una tonalidad de amarillo no tan brillante y tampoco tan intenso.



2. **MangosMalos:** se debe colocar las imágenes con mangos malos, son mangos con un color amarillo más oscuro, y manchas negras, lo que indica que el mangos ha dejado de estar fresco, a cierto grado de descomposición, y no apto para el consumo humano

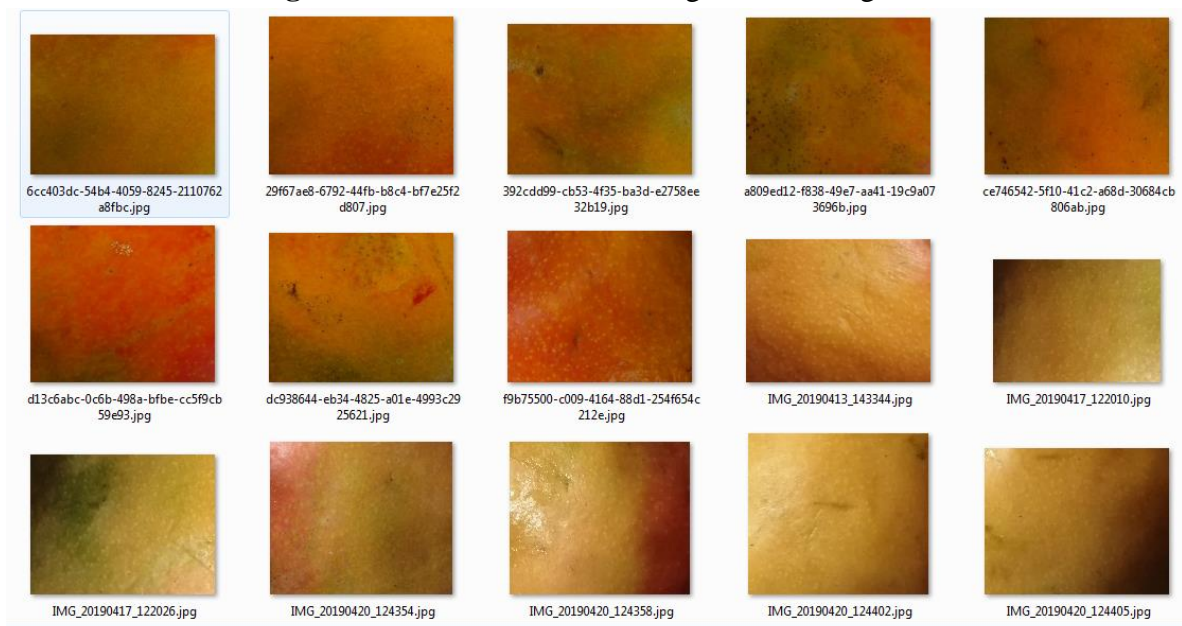


3. **MangosVerdes:** se debe colocar las imágenes con mangos verdes, estos mangos se caracterizan por su tonalidad verdosa en casi todo el cuerpo del mango, este color tan peculiar indica que el mangos está verde, que aún le falta tiempo para madurar completamente, estando en un estado no adecuado para el consumo.

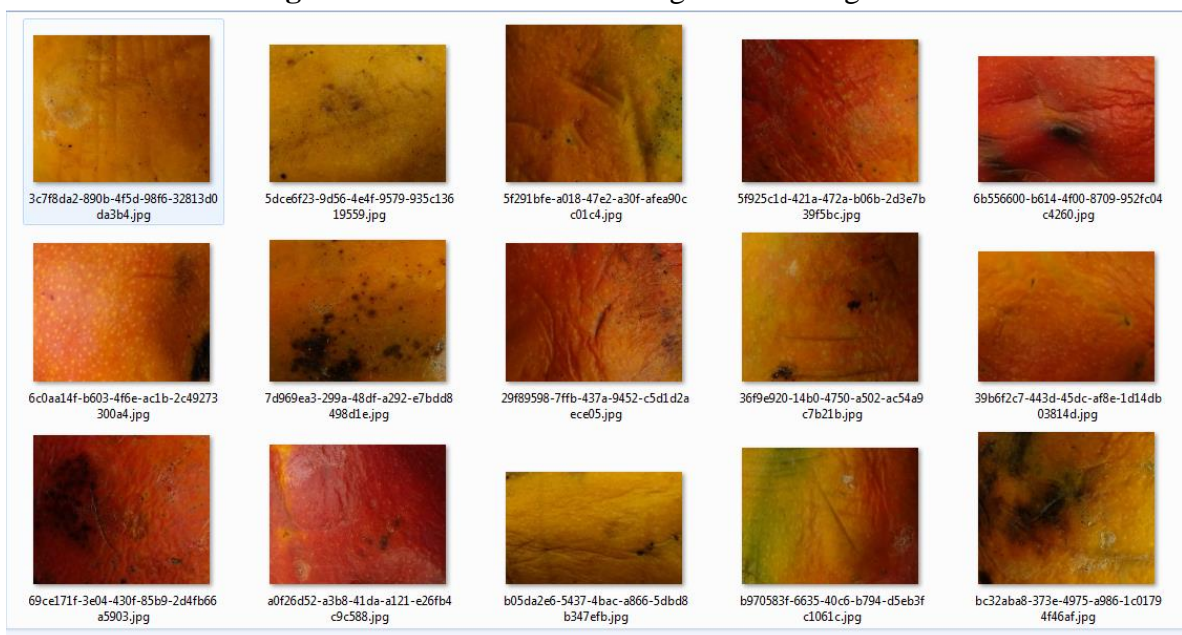




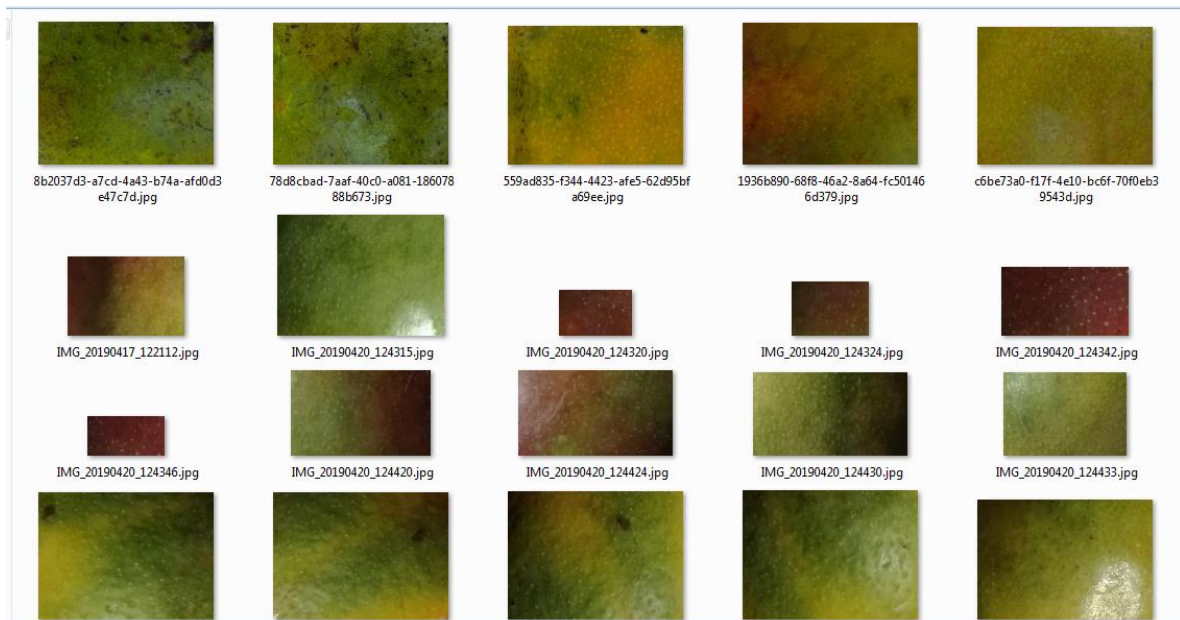
#### 4. RecortesMangosBuenos: recortes de las imágenes de mangos buenos



#### 5. RecortesMangosMalos: recortes de las imágenes de mangos malos

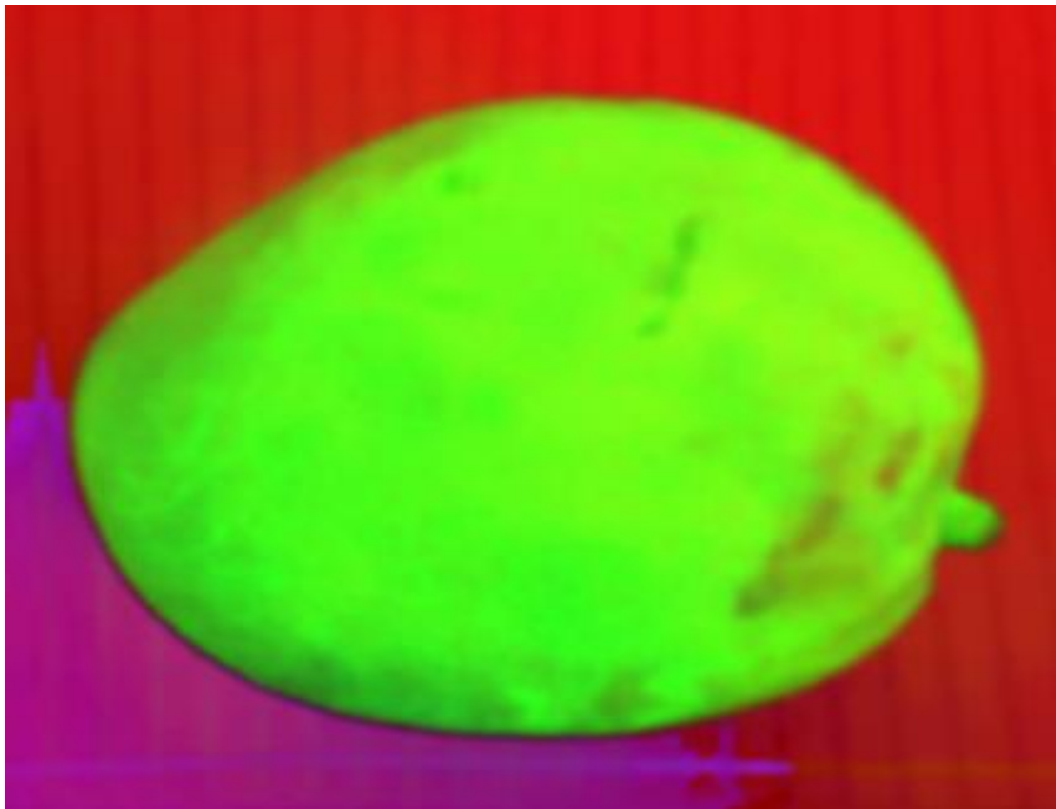


## 6. **RecortesMangosVerdes**: recortes de las imágenes de mangos verdes



## 7. **RecorteMango.py**: programa que extrae un recorte de un mangos, de una foto, de los directorios mencionados, busca los colores amarillos en los rangos rgb

Al mango se le cambia haciendo que todo lo que no sea amarillo se vuelva rojo.

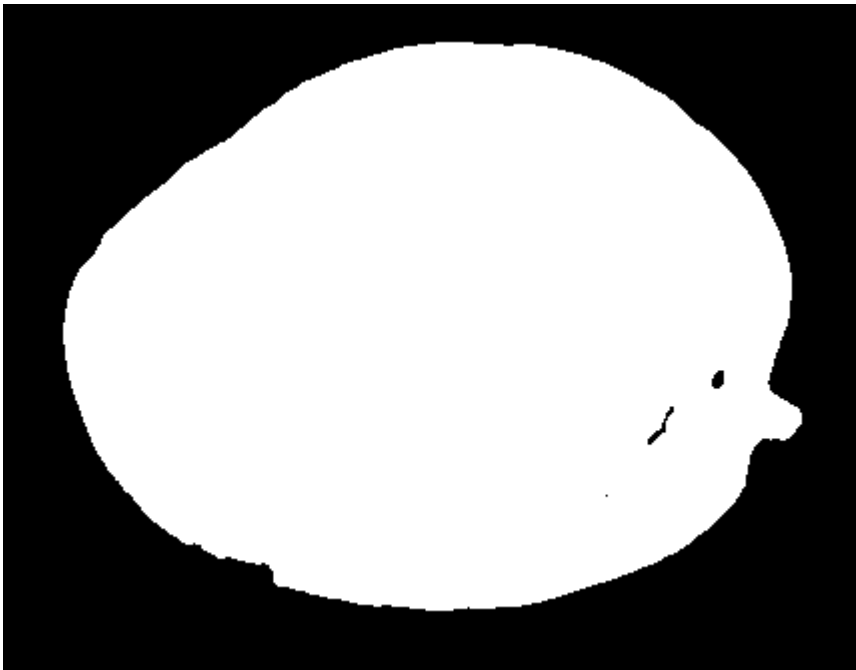


luego se detecta las partes amarillas de esta imagen en dos capas una para amarillos fuertes y otra para amarillos débiles

```
imagen_azul = cv2.GaussianBlur(imagen2, (7, 7), 0)
min_amarillo = np.array([0, 155, 25])
max_amarillo = np.array([256, 256, 256])

min_amarillo2 = np.array([180, 100, 120])
max_amarillo2 = np.array([248, 234, 239])
mascara2 = cv2.inRange(imagen_azul, min_amarillo2, max_amarillo2)
```

Estas dos imágenes se unen para dar como resultado

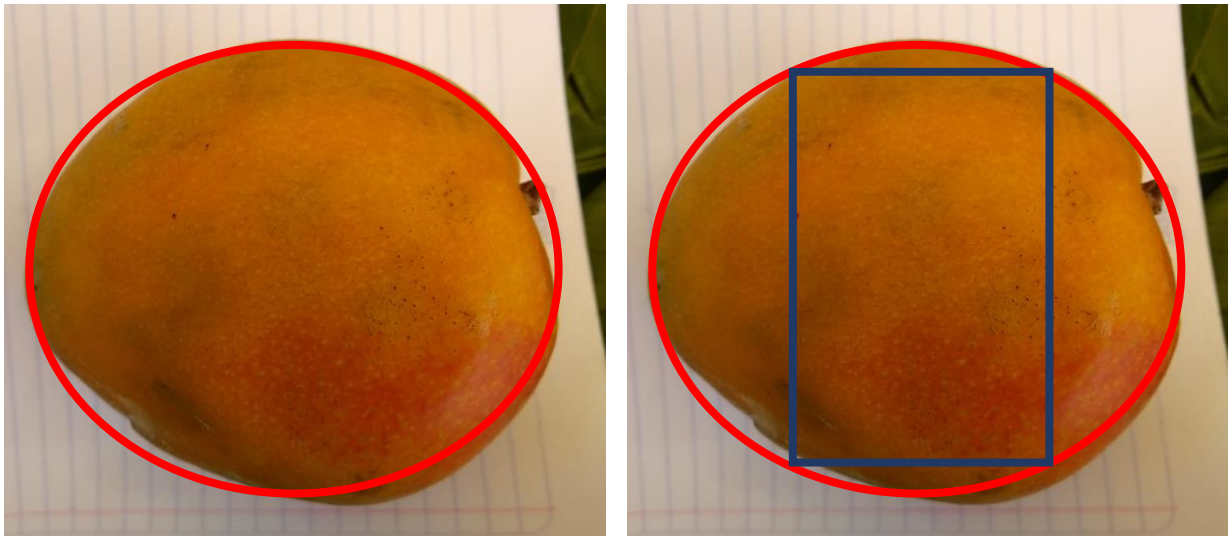


Esto nos genera la imagen en negativo del mangos, como se puede observar sin reflejo esta imagen se le envía a la función de

```
elipse = cv2.fitEllipse(contorno)
```

Esta línea de código dibuja una elipse sobre el contorno blanco

Las coordenadas que se obtienen se dibujan sobre la imagen original



Luego se obtiene de la función de la elipse los datos de las coordenadas necesarias para poder dibujar un cuadrado sobre la imagen

Estas coordenadas son las que dan el resultado final del recorte que se obtiene y se almacena en la carpeta recortes mencionadas anteriormente

Las imágenes se obtienen directamente de la carpeta.

```
rDirectorio("MangosBuenos", "RecortesMangosBuenos", listdir("./MangosBuenos"))
rDirectorio("MangosMalos", "RecortesMangosMalos", listdir("./MangosMalos"))
rDirectorio("MangosVerdes", "RecortesMangosVerdes", listdir("./MangosVerdes"))
```

Este es el proceso realizado a la imagen para obtener el mango en una fotografía ingresada por el usuario obteniendo resultados aceptables.

Los recortes almacenados tienen dimensiones distintas en el siguiente apartado modificamos el tamaño de la imagen para que sea estándar.

8. **DatosEntradaRNA:** programa que extrae los pixeles de las imágenes de los directorios, y los une en un archivo.csv. recorre la carpeta con las imágenes recortadas obteniendo su código rgb almacenando en un archivo csv

Cada imagen se le cambio la dimensión a **100 \* 50**

Antes de ser almacenadas los datos pasan a ser normalizados en un rango de 0 a 1, esto para mejorar la eficiencia del algoritmo al momento de entrenar, la función de transferencia fue la siguiente

$$\text{Normalizacion} = x/255$$

Donde x es uno de los 3 rgb obtenidos de un píxel esto retorna un valor entre 0 y 1

Dando como resultado el archivo datos-entrenamiento.csv

Es una muestra de los datos almacenados

```
0.827 0.392 0.388 0.823 0.380 0.396 0.831 0.364 0.384 0.854 0.356 0.352 0.870 0.345 0.321 0.854 0.317
0.258 0.984 0.396 0.274 0.992 0.411 0.282 0.992 0.419 0.274 0.984 0.411 0.270 0.980 0.407 0.274 0.988
0.431 0.278 0.976 0.435 0.274 0.976 0.439 0.278 0.984 0.443 0.298 0.988 0.439 0.325 0.984 0.435 0.309
0.988 0.419 0.247 0.968 0.427 0.231 0.976 0.447 0.227 0.980 0.454 0.235 0.980 0.454 0.235 0.992 0.454
0.247 0.929 0.349 0.243 0.952 0.356 0.258 0.960 0.356 0.247 0.956 0.352 0.243 0.960 0.356 0.239 0.968
0.980 0.419 0.262 0.992 0.439 0.278 0.988 0.435 0.262 0.988 0.435 0.258 0.984 0.431 0.258 0.988 0.435
0.282 0.984 0.454 0.278 0.964 0.435 0.254 0.980 0.443 0.266 0.984 0.427 0.258 1.0 0.431 0.274 0.980 0
0.203 0.972 0.454 0.203 0.980 0.458 0.2 0.972 0.447 0.184 0.972 0.431 0.180 0.980 0.431 0.192 0.796 0
0.250 0.972 0.396 0.243 0.988 0.415 0.270 0.988 0.419 0.286 0.984 0.415 0.282 0.992 0.427 0.290 0.984
0.450 0.282 0.980 0.450 0.278 0.980 0.447 0.298 0.980 0.443 0.309 0.988 0.454 0.309 0.988 0.454 0.309
0.992 0.435 0.282 0.992 0.435 0.270 0.984 0.435 0.254 0.988 0.431 0.243 0.984 0.431 0.239 0.992 0.443
0.266 0.913 0.349 0.266 0.933 0.364 0.266 0.949 0.356 0.262 0.952 0.349 0.247 0.964 0.364 0.254 0.960
```

Al final de cada línea se le agrego la salida esperada en la rna

```
rDirectorio("RecortesMangosBuenos", listdir("./RecortesMangosBuenos"), "0 1 0")
rDirectorio("RecortesMangosMalos", listdir("./RecortesMangosMalos"), "1 0 0")
rDirectorio("RecortesMangosVerdes", listdir("./RecortesMangosVerdes"), "0 0 1" )
```

9. **RNA.py**: entrena la red neuronal para reconocer la madurez de los mangos, obtiene el archivo TrainingData.csv almacenando los datos en una matriz, luego se le envía la librería de openlab para entrenamiento

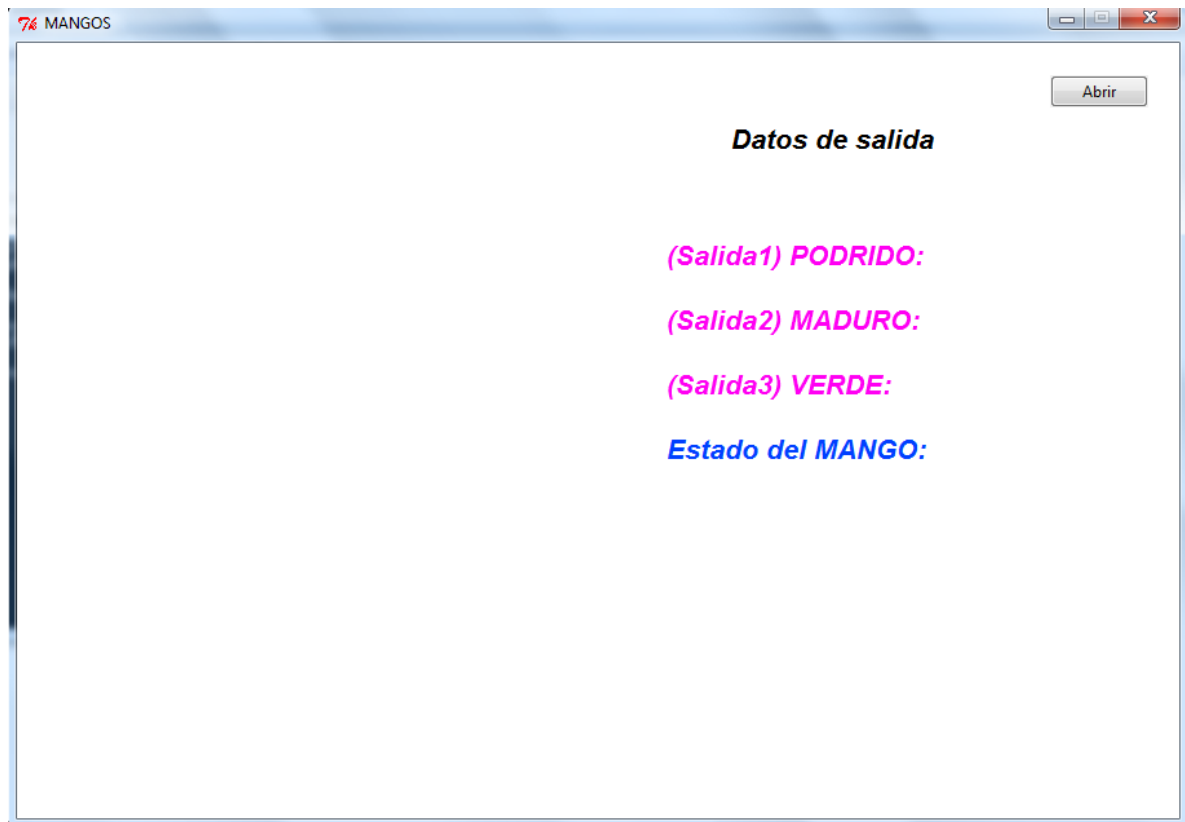
```
rna = nl.net.newff(maxmin, [ capa_entrada, capa_entrada, capa_oculta1, capa_salida])
```



```
Epoch: 1200; Error: 52.9982963776;  
Epoch: 1300; Error: 52.9981597727;  
Epoch: 1400; Error: 52.9979972416;  
Epoch: 1500; Error: 52.9978004765;  
Epoch: 1600; Error: 52.9975571708;  
Epoch: 1700; Error: 52.9972482659;  
Epoch: 1800; Error: 52.9968425412;  
Epoch: 1900; Error: 52.9962849345;  
Epoch: 2000; Error: 52.9954679227;  
Epoch: 2100; Error: 52.9941472014;  
Epoch: 2200; Error: 52.9916062313;  
Epoch: 2300; Error: 52.9841572065;  
Epoch: 2400; Error: 52.9895052519;  
Epoch: 2500; Error: 52.9693769689;
```

10. Interfaz: carpeta que contiene la aplicación para el usuario final

Para que el usuario final pueda interactuar y evaluar sus mangos se creo una aplicación de escritorio, este solo contiene un jtextdilgo para indicar la fotografía a evaluar y datos que indican la calidad del tomate, escribiendo en pantalla si está verde maduro o pasado




11. **.csv**: contiene los pixeles a evaluar de la(s) imagen(es).


Para abrir la imagen se le hace el mismo tratamiento que las imágenes para entrenar y luego de tener el recorte se abre el archivo generado por el entrenamiento enviando como parámetros los colores normalizados de la imagen abierta.

El archivo generado es Trained-RNA.tmt devolviendo tres salidas luego de esto se analizan las salidas lo que nos da los datos necesarios para determinar si el mango está verde maduro o pasado.

12. **.tmt**: red neuronal ya entrenada.

 Trained-RNA.tmt	26/04/2019 10:14 a...	Archivo TMT	52,173 KB
---	-----------------------	-------------	-----------

13. **EvaluacionFruta.py**: archivo para realizar la evaluación de la fruta, sino se desea utilizar la interfaz gráfica.

 EvaluacionFruta.py	26/04/2019 01:31 ...	Archivo PY	6 KB
--	----------------------	------------	------

## Bibliografía

### Páginas web:

- <http://guateinfoagro.blogspot.com/2012/06/el-cultivo-del-mango.html>  
<https://www.deguate.com/artman/publish/produccion-guatemala/produccion-de-mango-en-guatemala.shtml>
- opencv.org (2016). About. recuperdo de <http://opencv.org/about.html>
- pypi.python.org (2017). neurolab 0.3.5 Recuperado de <https://pypi.python.org/pypi/neurolab>
- es.wikipedia.org (2017). Propagacion hacia atras [https://es.wikipedia.org/wiki/Propagaci%C3%B3n\\_hacia\\_atr%C3%A1s#Regla\\_de\\_aprendizaje](https://es.wikipedia.org/wiki/Propagaci%C3%B3n_hacia_atr%C3%A1s#Regla_de_aprendizaje)