



# DiAgroDL v1.0

Plataforma WEB Inteligente Para La Identificación Y Diagnóstico  
De Problemas Fitosanitarios En Cultivos Agrícolas



Presenta:

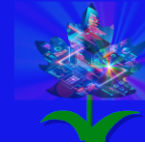
**Juan Pablo Ambrosio Ambrosio**

Posgrado:

**PSEI-CÓMPUTO APLICADO**

[ambrosio.juan@colpos.mx](mailto:ambrosio.juan@colpos.mx)

17 de julio de 2024



# ÍNDICE DE PRESENTACIÓN

## Marco teórico

- Redes neuronales (NN)
- Redes neuronales convolucionales (CNN)
- Capas convolucionales
- Capas de agrupamiento
- Capas de aplanamiento
- Funciones de activación
- Backpropagation
- Transferencia de aprendizaje

## Modelación

- Conjunto de datos
- Resultado de la modelación (VGG16)

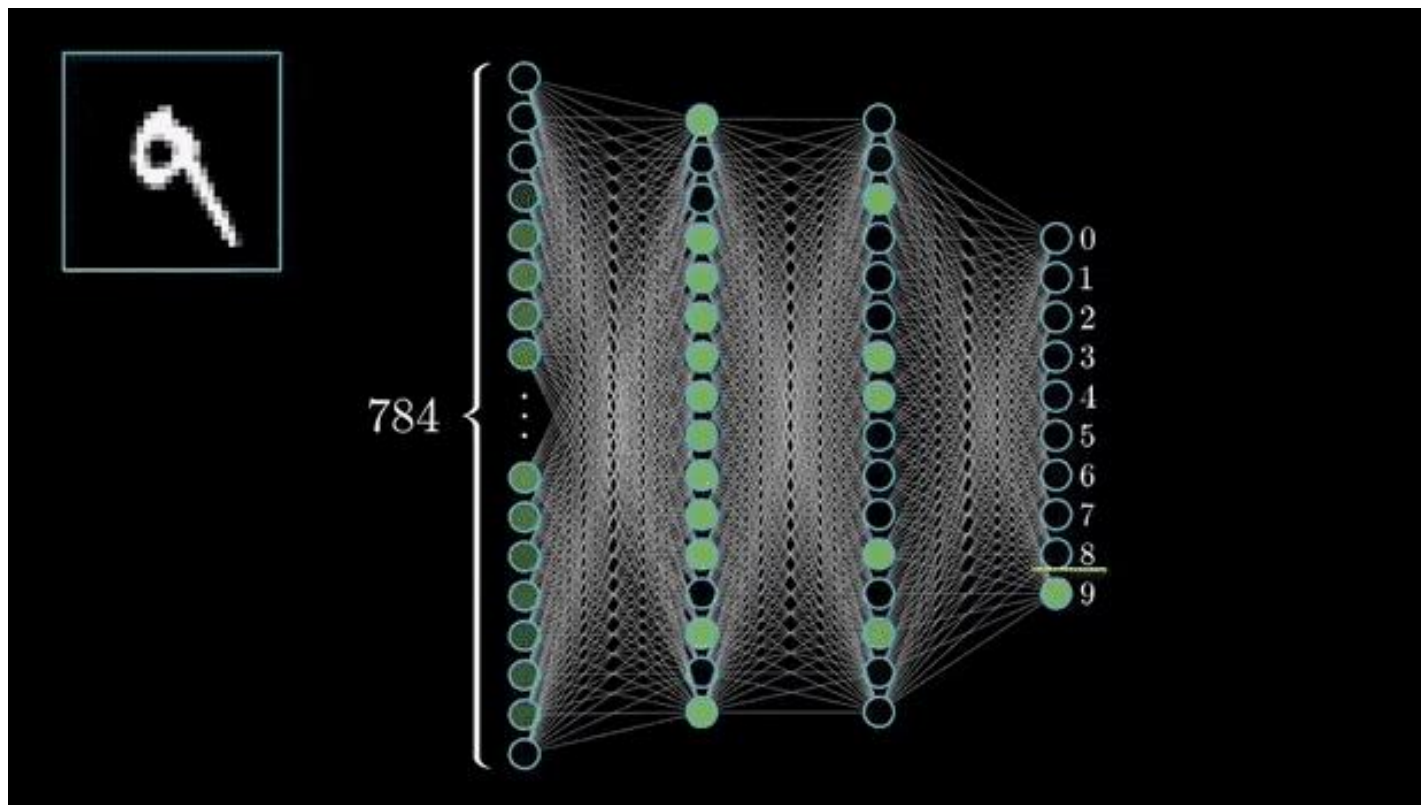
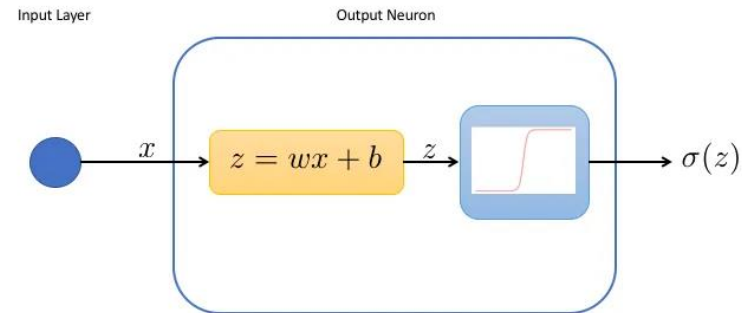
## Desarrollo tecnológico

- Sobre DiAgroDL v1.0
- ¿Cómo trabaja DiAgroDL v1.0?
- ¿Qué tecnología computacional usa DiAgroDL v1.0?
- Estructura del proyecto Django
- DiAgroDL v1.0 (video tutorial y manual de usuario)

## Conclusión

## Trabajos futuros

## ¿Qué son las redes neuronales?



## ¿Qué son las redes neuronales convolucionales (CNN)?

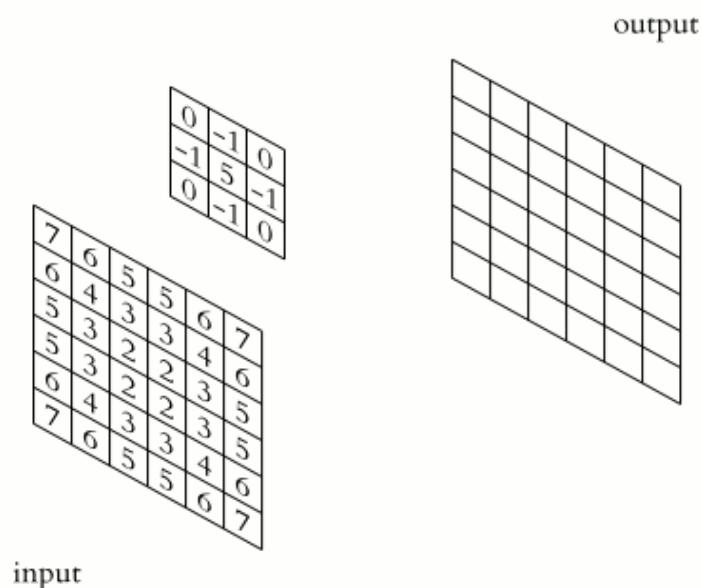
A diferencia de las redes neuronales estándar, las CNN tiene **capas convolucionales**. Estas capas realizan una operación matemática crítica conocida como **convolución**.

Este proceso implica la aplicación de filtros especializados conocidos como kernels, que recorren la imagen de entrada para aprender patrones visuales complejos.

### Capas convolucionales (Convolutional layers)

#### Kernels (Filtros)

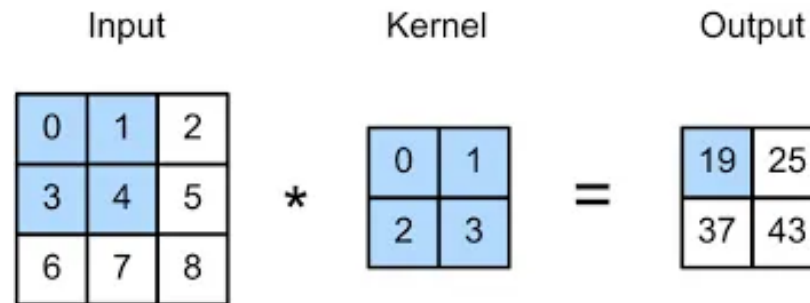
Son esencialmente pequeñas **matrices de números**. Estos filtros se desplazan por la imagen realizando multiplicaciones por elementos con la parte de la imagen que cubren, extrayendo características como **bordes, texturas y formas**.





## Pero... ¿Cuál es la operación de convolución?

La operación de convolución consiste en multiplicar los valores del Kernel por los valores de los píxeles originales de la imagen y sumar los resultados.



Mapa de características

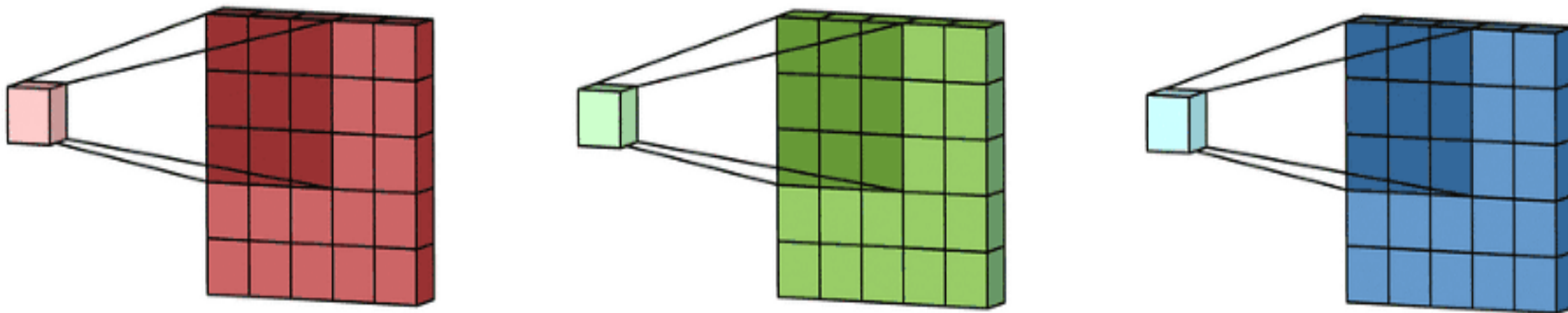
$$(0 \times 0) + (1 \times 1) + (3 \times 2) + (4 \times 3) = 19$$

$$(1 \times 0) + (2 \times 1) + (4 \times 2) + (5 \times 3) = 25$$

$$(3 \times 0) + (4 \times 1) + (6 \times 2) + (7 \times 3) = 37$$

$$(4 \times 0) + (5 \times 1) + (7 \times 2) + (8 \times 3) = 43$$

## Convolución en imágenes RGB



1.- Channels

2.- Stride

3.- Padding



## Capas de agrupamiento (Pooling layers)

Imagina que tienes una **imagen grande** y quieres hacerla **más pequeña**, pero conservando todas las características importantes, como los bordes y los colores. La capa de agrupación opera de forma independiente en cada mapa de características de la entrada. La redimensión espacial se realiza utilizando el **máximo** o la **media** de los valores de una ventana deslizada sobre los datos de entrada.

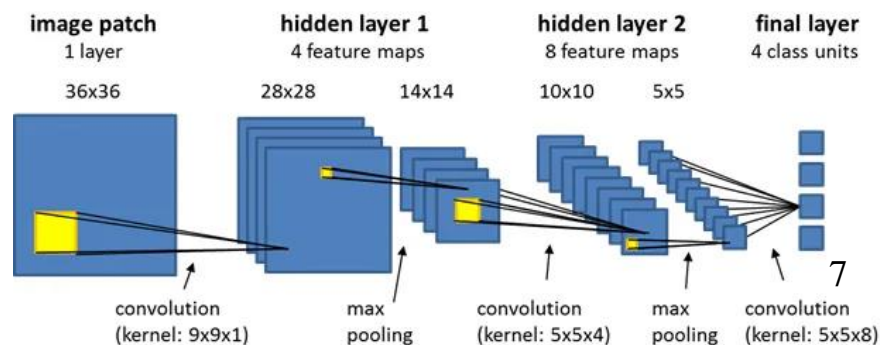
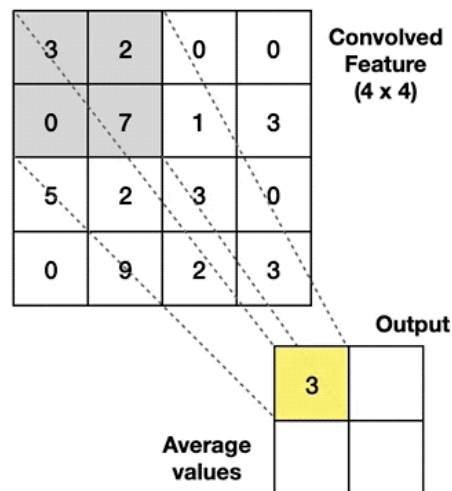
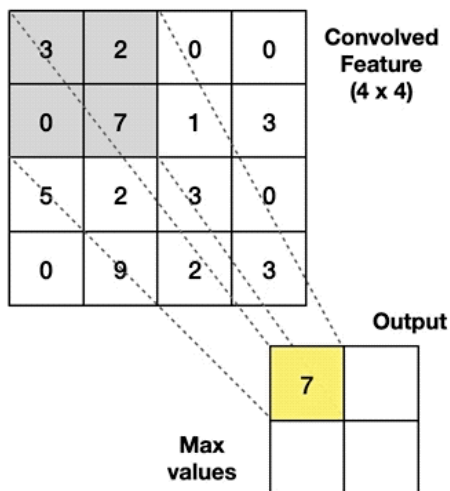
### Max Pooling

Take the **highest** value from the area covered by the kernel

### Average Pooling

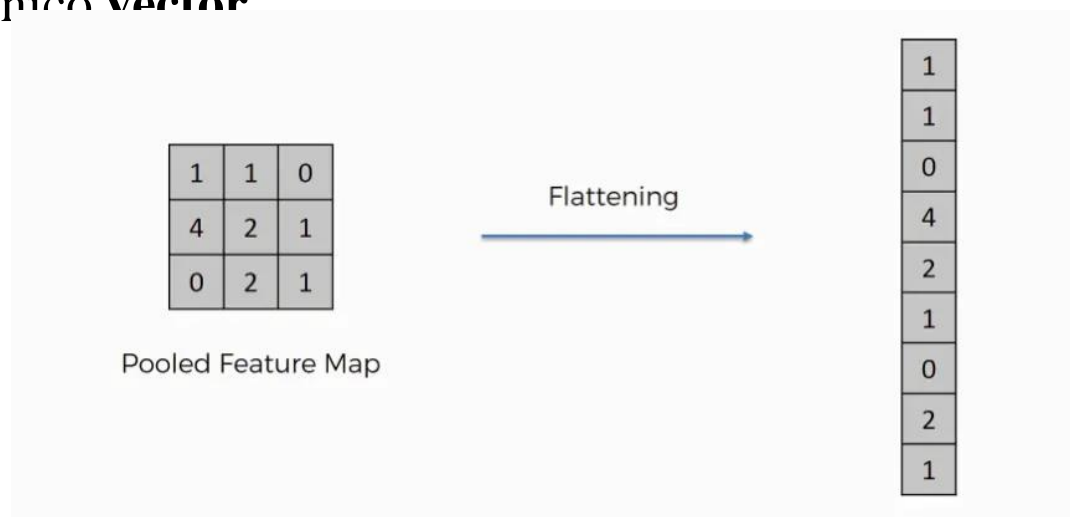
Calculate the **average** value from the area covered by the kernel

Example: Kernel of size 2 x 2; stride=(2,2)



## Capas de aplanamiento (Flattening layers)

Imagine que tiene una cuadrícula de datos (como los píxeles de un mapa de **características**) y desea alinear todos los puntos de la cuadrícula en una única línea larga. Eso es lo que hace el aplanamiento. Toma todo el mapa de características y lo reorganiza en un único **vector**



### Integración de características

Al aplanar los mapas de características en un vector, la red puede **integrar las características** distribuidas espacialmente **extraídas** para tareas como la clasificación.

### Compatibilidad con las capas densas

Las capas totalmente conectadas (capas densas) están diseñadas para operar con datos **unidimensionales**, por lo que el aplanamiento es un paso necesario para pasar de los tensores multidimensionales producidos por las capas convolucionales al formato requerido por las capas<sup>8</sup> densas.



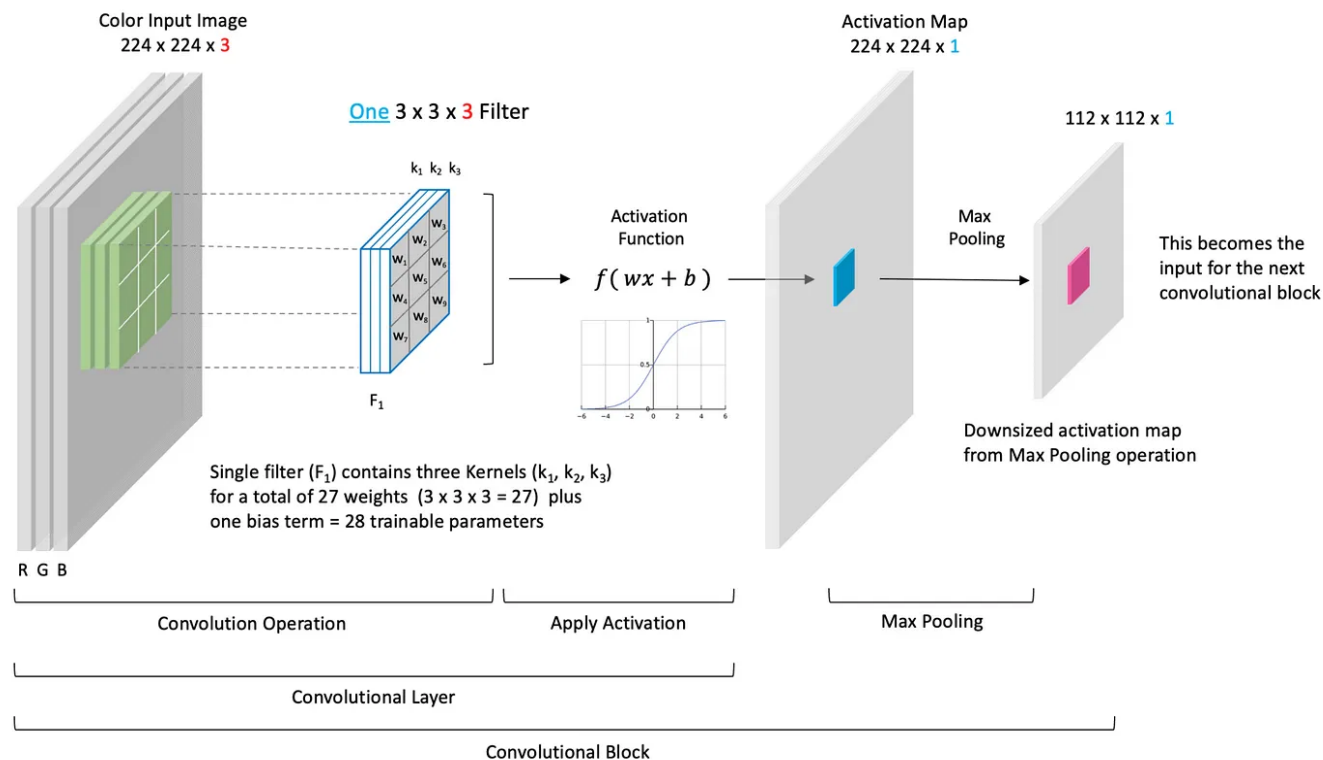
## Funciones de activación

Las funciones de activación son indispensables, de lo contrario, estaríamos creando un modelo lineal muy grande.

Al igual que en las RN simples, también necesitamos estos términos no lineales en las ConvNets. Sin embargo, **no todas las capas** que hemos visto **tienen una función** de activación.

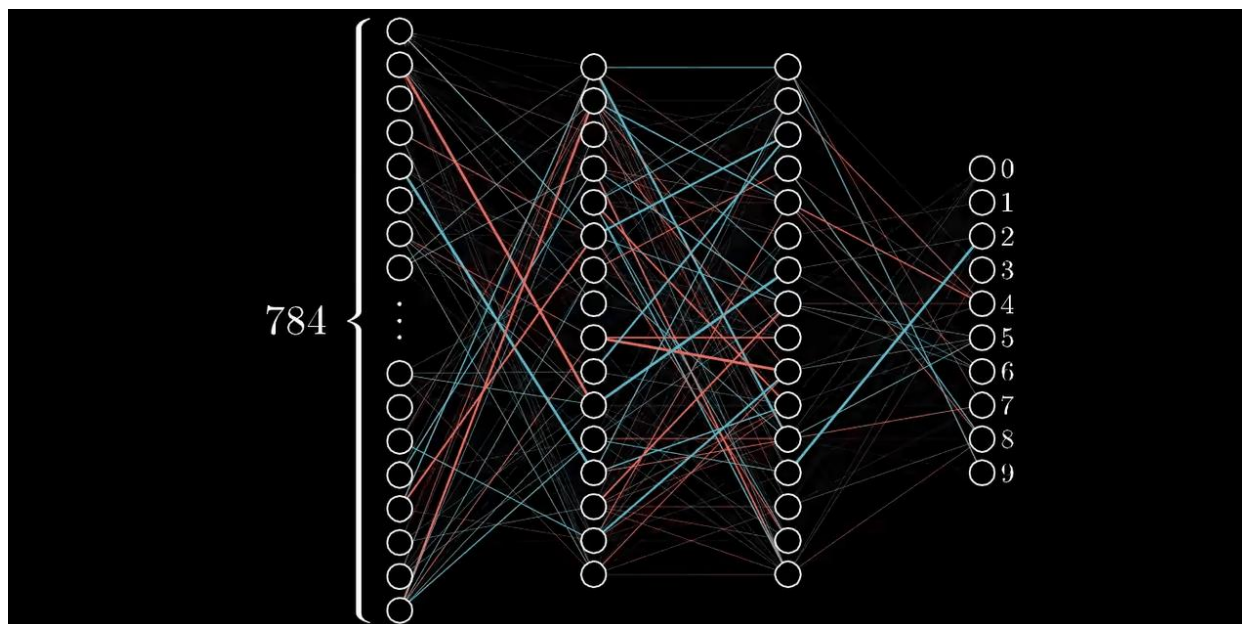
En la parte de **extracción de características**, las **activaciones** estarán en las capas **convolucionales**. El proceso es bastante sencillo, después de cada operación de convolución se multiplica el resultado por una función de activación.

- RELU
- LEAK RELU



- SIGMOID
- TANH
- SOFTMAX

# Backpropagation



El término **Backpropagation**, abreviatura de "**backward propagation of errors**", es un algoritmo de aprendizaje supervisado que se utiliza para **minimizar los errores** en las predicciones realizadas por redes neuronales.

En principio, la retropropagación es una aplicación de reglas en cadena (**chain**) que puede utilizarse para calcular gradientes (**gradient**) de funciones de pérdida en relación con los parámetros del modelo.

El mecanismo funciona en dos fases principales:

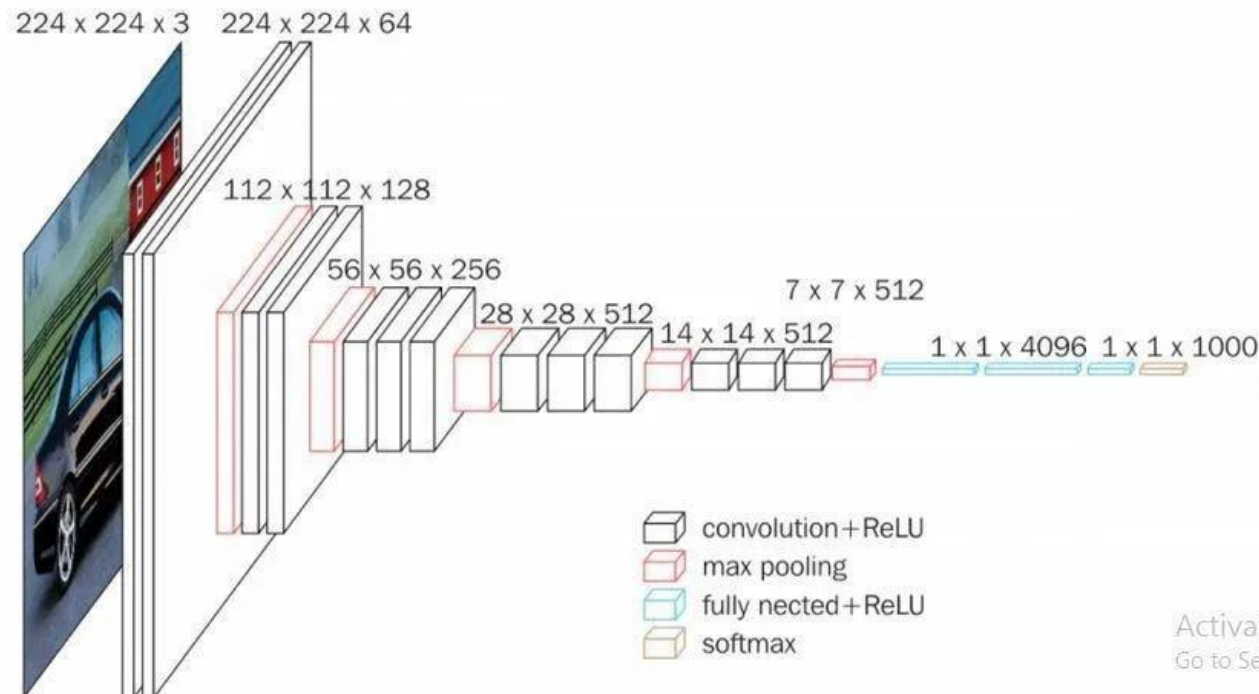
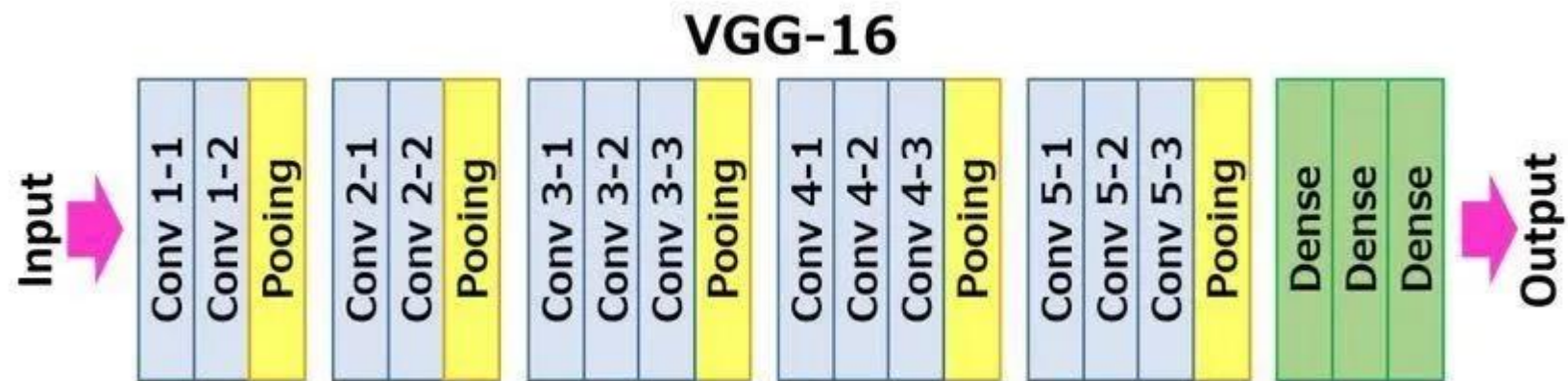
- 1.- **Forward pass**
- 2.- **Backward pass**

# Transfer learning

## VGG16

## RESNET

## MOBILENET







## Conjunto de datos (Kaggle)

Bacterial\_spot  
(1,701; 212; 214)



Septoria\_leaf\_spot  
(1,416; 177; 178)



Tomato\_healthy  
(1,272; 159; 160)



Yellow\_leaf\_curl\_virus  
(2,566; 320; 322)



Early\_blight  
(800; 100; 100)



Spider\_mites\_two\_spotted  
(1,340; 167; 169)



Tomato\_mosaic\_virus  
(298; 37; 38)



Leaf\_mold  
(761; 95; 96)



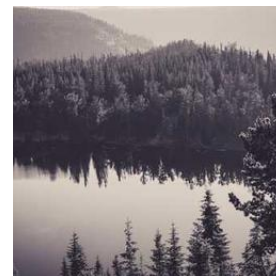
Late\_blight  
(1,526; 190; 192)



Target\_spot  
(1,123; 140; 141)



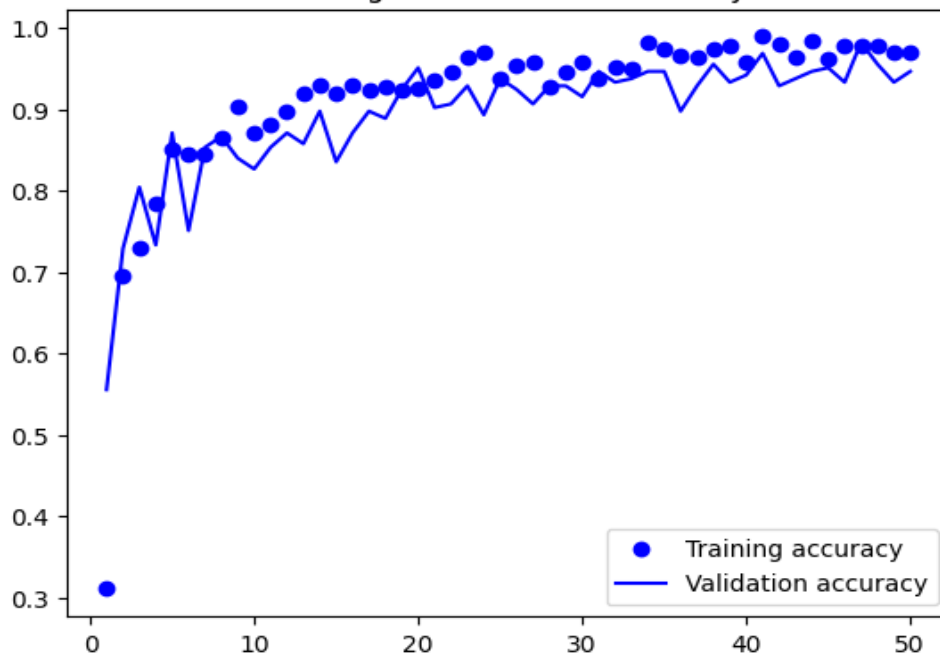
Unknown  
(800; 100; 100)



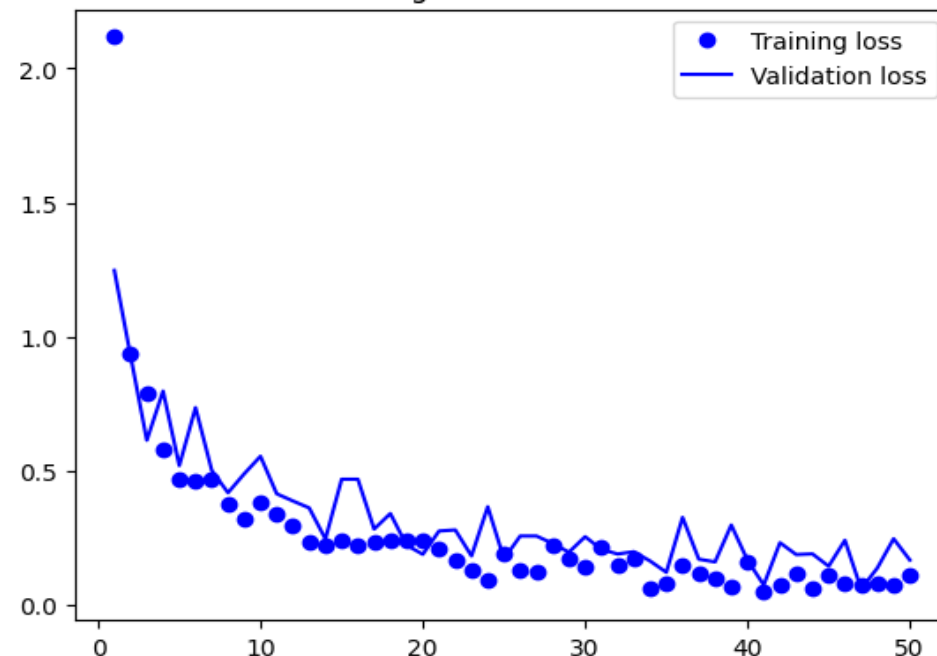
Hughes P., D. and M. Salathé.  
2015. An open access repository  
of images on plant health to  
enable the development of  
mobile disease diagnosis.  
arXiv:1511.08060.

# Resultado de la modelación (VGG16)

Training and validation accuracy



Training and validation loss



Epoch 41/50

30/30 [=====] - 68s 2s/step - loss: 0.0454 - accuracy: 0.9893

- val\_loss: 0.0733 - val\_accuracy: 0.9689

Layer (type)	Output Shape	Param #	Trainable
input_2 (Input Layer)	[(None, 256, 256, 3)]	0	Y
rescaling (Rescaling)	(None, 256, 256, 3)	0	Y
vgg16 (Functional)	(None, 8, 8, 512)	14,714,688	Y
flatten (Flatten)	(None, 32768)	0	Y
dense (Dense)	(None, 525)	17,203,725	Y
dropout (Dropout)	(None, 525)	0	Y
dense (Dense)	(None, 11)	5,786	Y
Total params:	31,924,199 (121.78 MB)		
Trainable params:	24,288,935 (92.65 MB)		
Non-trainable params:	7,635,264 (29.13 MB)		

# Resultado de la modelación (VGG16)

Transfer Learning (VGG16) with Fine-Tuning

True Label		Bacterial_spot	Early_blight	Late_blight	Leaf_mold	Septoria_leaf_spot	Spider_mites_two_spotted	Target_spot	Tomato_healthy	Tomato_mosaic_virus	Unknown	Yellow_leaf_curl_virus
	Bacterial_spot	204	1	0	0	5	0	2	0	0	0	2
	Early_blight	3	87	5	1	1	2	1	0	0	0	0
	Late_blight	1	5	165	7	7	2	4	0	0	1	0
	Leaf_mold	0	0	0	94	1	1	0	0	0	0	0
	Septoria_leaf_spot	0	0	0	2	175	0	0	0	1	0	0
	Spider_mites_two_spotted	0	0	1	0	0	157	10	0	1	0	0
	Target_spot	1	0	0	1	4	3	132	0	0	0	0
	Tomato_healthy	0	0	0	0	0	1	1	158	0	0	0
	Tomato_mosaic_virus	0	0	0	0	0	0	0	0	38	0	0
	Unknown	0	0	0	0	0	0	0	0	0	100	0
	Yellow_leaf_curl_virus	2	0	0	2	0	2	1	0	0	0	315
	Predicted Label	Bacterial_spot	Early_blight	Late_blight	Leaf_mold	Septoria_leaf_spot	Spider_mites_two_spotted	Target_spot	Tomato_healthy	Tomato_mosaic_virus	Unknown	Yellow_leaf_curl_virus



## Resultado de la modelación (VGG16)

**PG\_test: 95.03%**

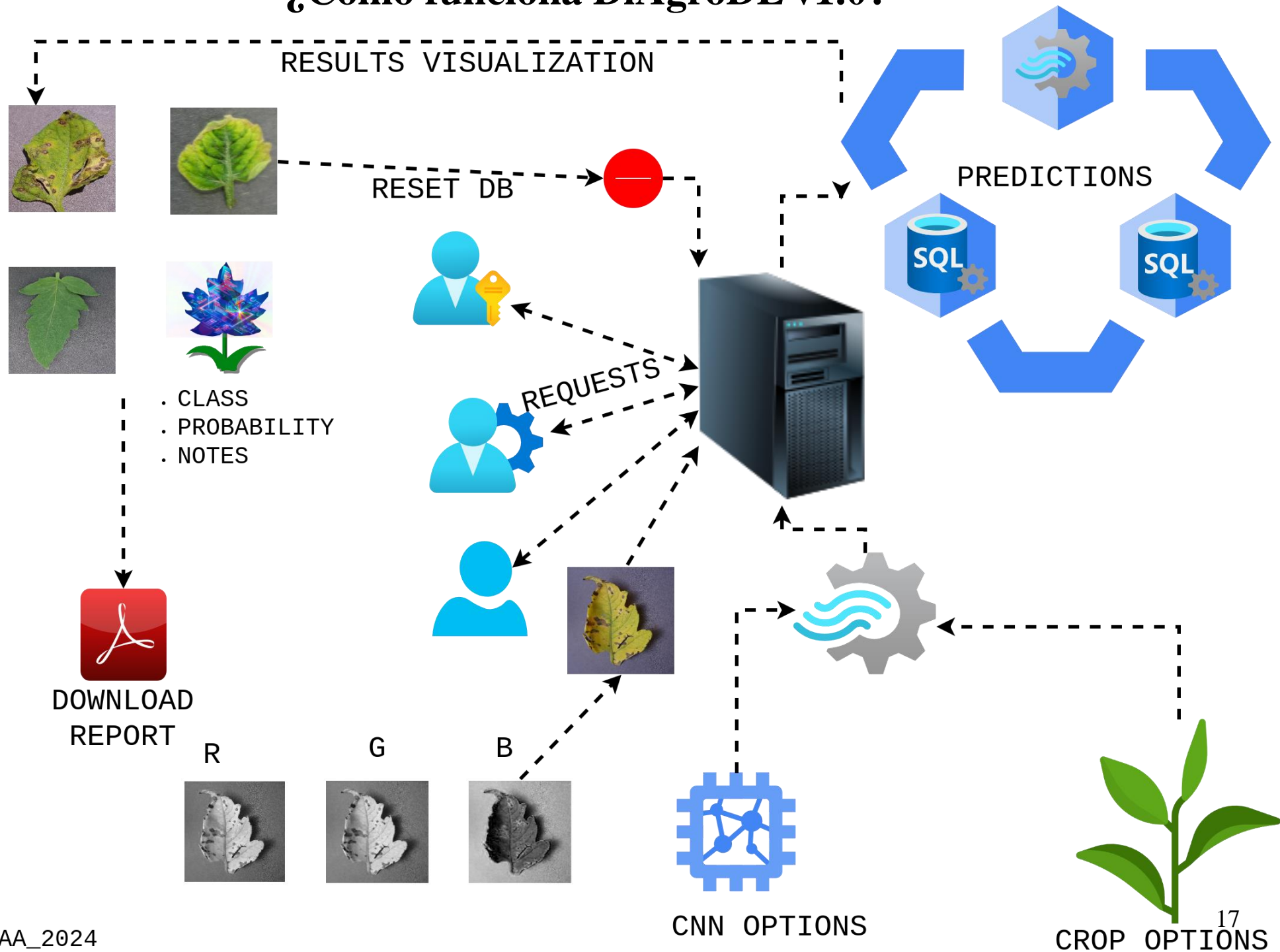
	Precision	Recall	F-Score	Support
Unknown	0.99	1.00	1.00	100
Tomato_healthy	1.00	0.99	0.99	160
Yellow_leaf_curl_virus	0.99	0.98	0.99	322
Tomato_mosaic_virus	0.95	1.00	0.97	38
Bacterial_spot	0.97	0.95	0.96	214
Septoria_leaf_spot	0.91	0.98	0.94	178
Spider_mites_two_spotted	0.93	0.93	0.93	169
Leaf_mold	0.88	0.98	0.93	96
Late_blight	0.96	0.86	0.91	192
Target_spot	0.87	0.94	0.90	141
Early_blight	0.94	0.87	0.90	100

## ACERCA DE DiAgroDL v1.0

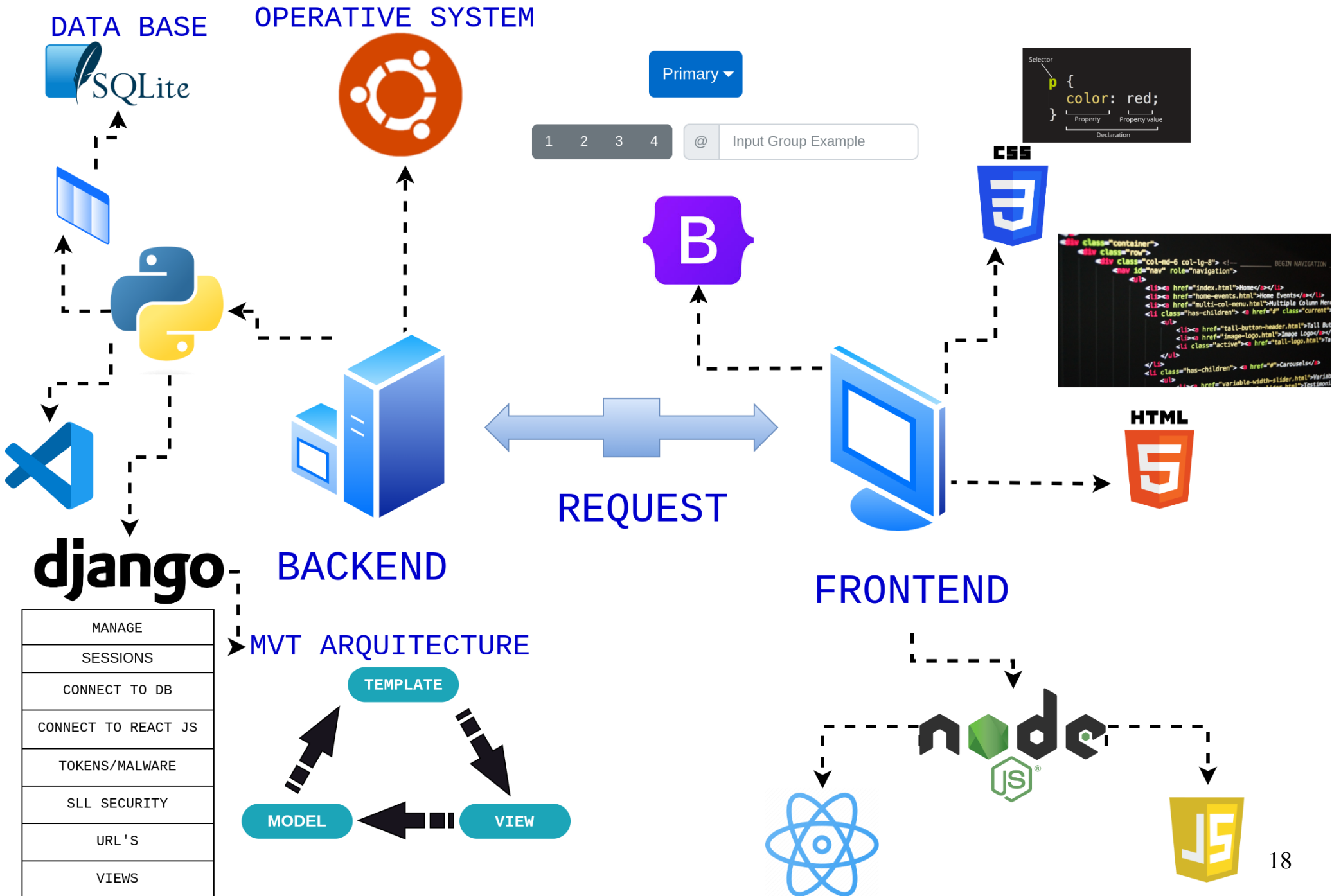
DiAgroDL v1.0 es una aplicación WEB desarrollada por el equipo de investigación del posgrado en SEI – CÓMPUTO APLICADO del COLEGIO DE POSTGRADUADOS campus Montecillo. Es un sistema intuitivo y sencillo que permite diagnosticar nueve enfermedades en hojas de tomate y la planta testigo, en este caso una hoja completamente sana. Las enfermedades que el sistema considera son: 'Bacterial\_spot', 'Early\_blight', 'Late\_blight', 'Leaf\_mold', 'Septoria\_leaf\_spot', 'Spider\_mites', 'Target\_spot', 'Tomato\_healthy', 'Tomato\_mosaic\_virus', y 'Yellow\_leaf\_curl\_virus'. La versión 1.0 de DiAgroDL puede realizar el diagnóstico mediante el uso de tres modelos de redes neuronales convolucionales (CNN) basado en las arquitecturas: VGG16, RESNET y Mobile NET.

Enfermedades en la hoja del tomate: 'Mancha bacteriana', 'Tizón temprano', 'Tizón tardío', 'Moho de la hoja', 'Mancha foliar por Septoria', 'Araña roja', 'Mancha diana', 'Tomate sano', 'Virus del mosaico del tomate', y 'Virus del rizado amarillo de la hoja'.

# ¿Cómo funciona DiAgroDL v1.0?



# ¿Qué tecnología computacional usa DiAgroDL v1.0?



## Estructura del proyecto Django

The screenshot shows the Visual Studio Code interface with the Django project structure on the left and the `Welcome.js` component on the right.

**EXPLORER (Left Panel):**

- WEB\_SITE
  - .vscode
  - Additional
  - muestras
  - Scripts\_py
  - tomato\_project
    - AppDjango
      - \_\_pycache\_\_
      - migrations
      - models
      - \_\_init\_\_.py
      - admin.py
      - apps.py
      - models.py
      - serializers.py
      - tests.py
      - urls.py
      - views\_1.py
      - views.py
    - media
    - react\_app
      - build
      - node\_modules
      - public
      - src
    - .gitignore

**JS Welcome.js (Right Panel):**

```
export class Welcome extends Component{
  render(){
    return(
      <div className="m-t5 dflex justify-content-left">
        <h3></h3>
        <Alert key='primary' variant='primary'>
          Bienvenido a DiAgroDL, una aplicación WEB destinado a la identificación, y
        </Alert>
        <h3></h3>
        <Alert key='primary1' variant="primary">
          Aquí puedes revisar el video tutorial y el manual de usuario de DiAgroDL
        </Alert>
        <h3></h3>
        <Download pdf/>
        <h3></h3>
        <h3></h3>
        <div className="ratio ratio-16x9">
          <iframe
            src="https://www.youtube.com/embed/j0-DLGzp9Fc"
            title="DIAGRODL BY COLEGIO DE POSTGRADUADOS - CÓMPUTO APLICADO"
            allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope;
            allowFullScreen
          ></iframe>
        </div>
        <h3></h3>
      </div>
    )
  }
}
```


**TERMINAL (Bottom Panel):**

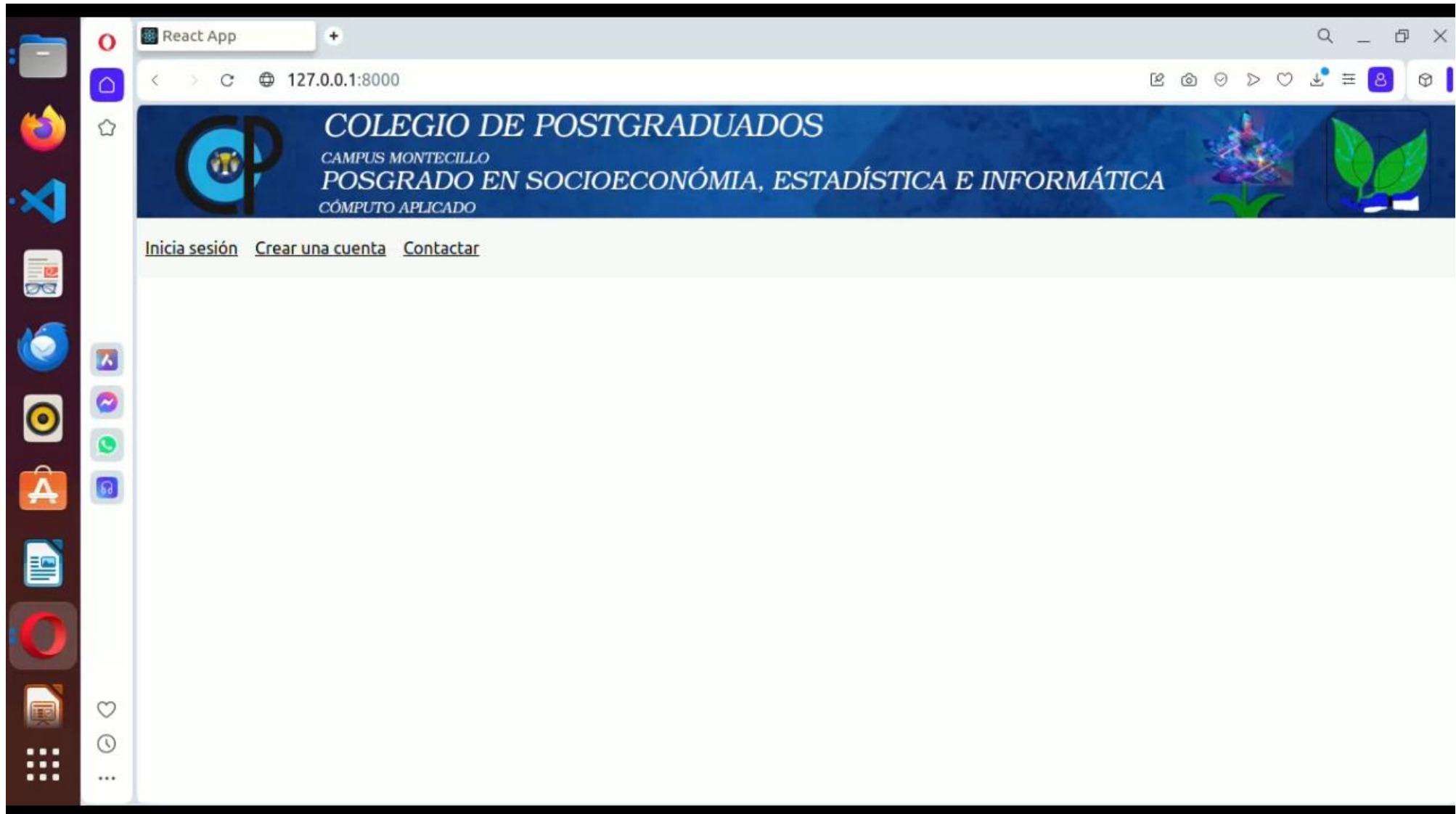
```
tomato venvjuan@juan-Latitude-3490:~/Documents/2024/COLPOS/TOMATO_DISEASE/WEB_SITE/tomato_project$
```



Video tutorial



Manual de usuario 





## **Conclusión:**

- La tecnología informática Web IA desarrollada (DiAgroDL v1.0) permite con alta precisión realizar el diagnóstico de nueve enfermedades en el cultivo de jitomate a partir de imágenes digitales de hojas y de modelos de aprendizaje profundo.
- La arquitectura de código abierto de DiAgroDL permite incorporar nuevos modelos IA y otros cultivos y enfermedades; para facilitar la identificación de problemas fitosanitarios en cultivos agrícolas desde un dispositivo móvil o PC.
- DiAgroDL v1.0 es una contribución tecnológica para reducir los costos y el tiempo en el diagnóstico de enfermedades en cultivos agrícolas.

## **Trabajos futuros:**

- Establecer colaboración con instituciones y expertos en fitosanidad para incorporar nuevos cultivos y enfermedades asociadas en diferentes regiones o zonas agrícolas de México.
- Transferir y evaluar el uso de la tecnología IA desarrollada en regiones específicas en colaboración con el sector agrícola y rural.

# !Gracias!

 [ambrosio.juan@colpos.mx](mailto:ambrosio.juan@colpos.mx)

 [www.linkedin.com/in/juan-pablo-ambrosio-ambrosio](https://www.linkedin.com/in/juan-pablo-ambrosio-ambrosio)

 <https://github.com/JPAAPSEICOA>