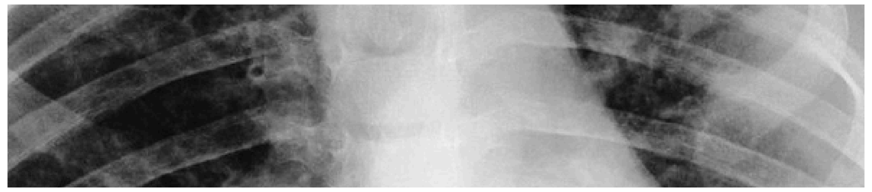
COMPUTER VISION PROJECT



NOT A DOCTOR

AI-POWERED CHEST X-RAY ANALYSIS TOOL

12.2024

TITOUAN G.

Computer Vision (CV) is a field of artificial intelligence that enables computers to interpret, analyze, and understand visual data from the world.

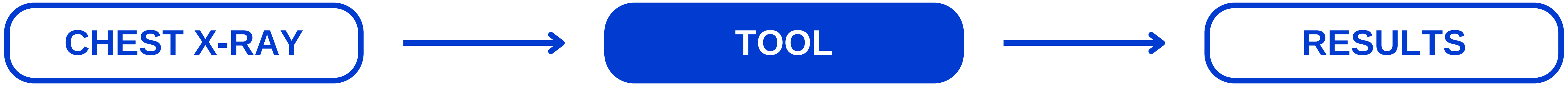
By mimicking human vision, computer vision systems process images or videos to extract meaningful information, identify patterns, and make decisions.

From facial recognition to autonomous vehicles, CV applications are transforming industries by automating tasks that traditionally required human sight.

Using advanced algorithms, machine learning, and neural networks, these systems can detect objects, track movements, and even recognize emotions with remarkable precision.

This technology empowers businesses, researchers, and developers to push the boundaries of innovation, creating solutions that enhance efficiency, accuracy, and creativity across a wide range of sectors.

“Not a doctor” is a project aimed to use computer vision for medical purpose. This tool analyzes chest X-ray images using machine learning algorithms to identify potential patterns associated with specific lung conditions.



CHEST X-RAY → TOOL → RESULTS

INTRODUCTION

Data in machine learning serves as the foundation for training algorithms, enabling them to identify patterns, make predictions, and generalize to new, unseen scenarios. In my project, data processing involves several key steps:

- **Data acquisition**

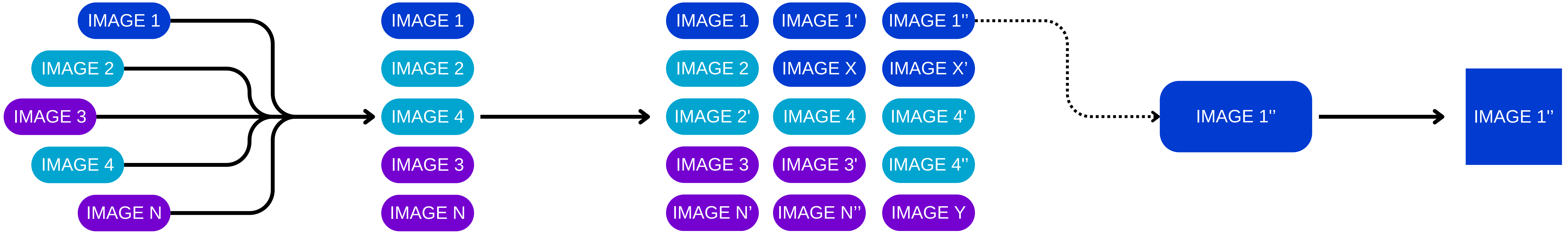
  For this project, I used a pre-built dataset from kaggle. The data consists of x-ray images, classified into 3 categories: NORMAL, COVID and VIRAL PNEUMONIA.

- **Data augmentation**

  Data Augmentation is a technique used to artificially increase the size and diversity of a dataset by applying transformations like rotation, flipping, cropping, or color adjustments to existing data.

- **Preprocessing**

  This step consists of converting images into a model-readable format and normalize them.



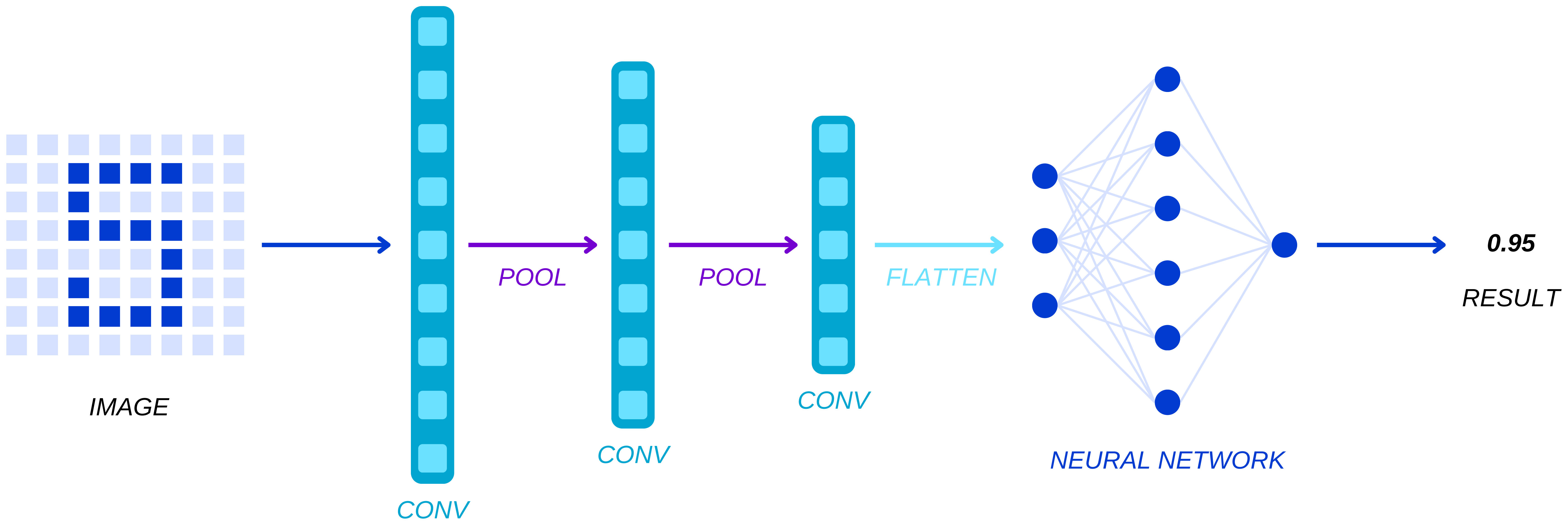
DATA

Deep Learning has revolutionized Computer Vision by enabling models to automatically learn complex features from data, surpassing traditional methods. At its core, deep learning leverages neural networks with multiple layers to process and analyze visual information.

Key architectures include:

- **Convolutional Neural Networks** (CNNs) for image recognition
- **Recurrent Neural Networks** (RNNs) for sequential data
- **Transformers**, which excel in handling large-scale vision tasks and contextual understanding.

For this project, I used a Convolutional Neural Network:

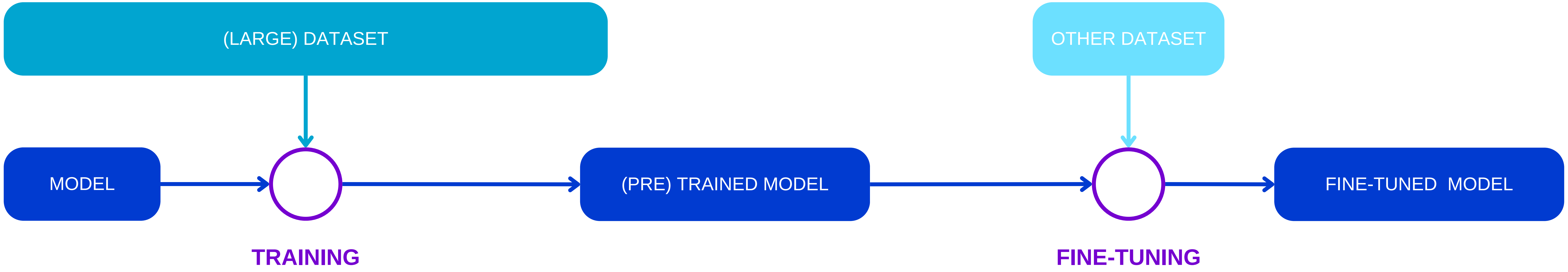


DEEP LEARNING

Transfer Learning is a machine learning technique where a pre-trained model, developed for one task, is adapted to solve a different but related task.

In Computer Vision, it often involves using models trained on large datasets as a starting point.

By reusing learned features such as edges or textures, transfer learning significantly reduces the amount of data and computational resources needed. It accelerates development and improving performance on specialized tasks.



Pre-trained models are readily available online or through libraries like TensorFlow or PyTorch.

I started by fine-tuning several pre-trained models on my dataset, adapting their learned features to the specific requirements of my project.

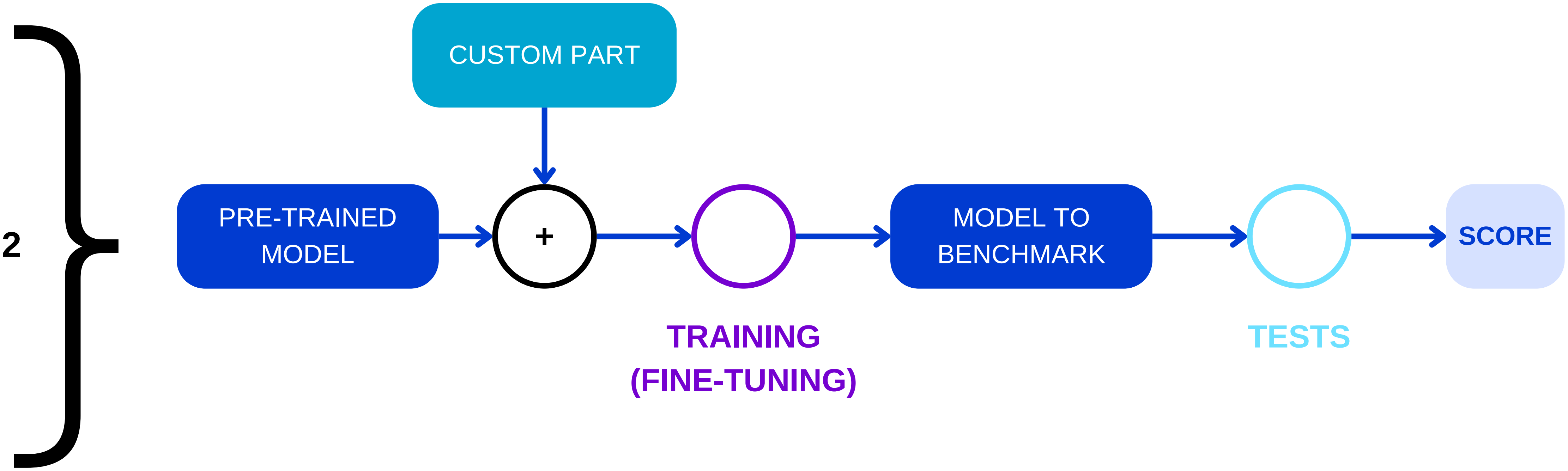
Next, I conducted a benchmark by evaluating each model's performance using their loss and accuracy metrics to assign a score. I then ranked the models based on their scores and selected the highest-performing one for my project.

SCORE = ACCURACY / (NUMBER OF PARAMETERS * LOSS) * C

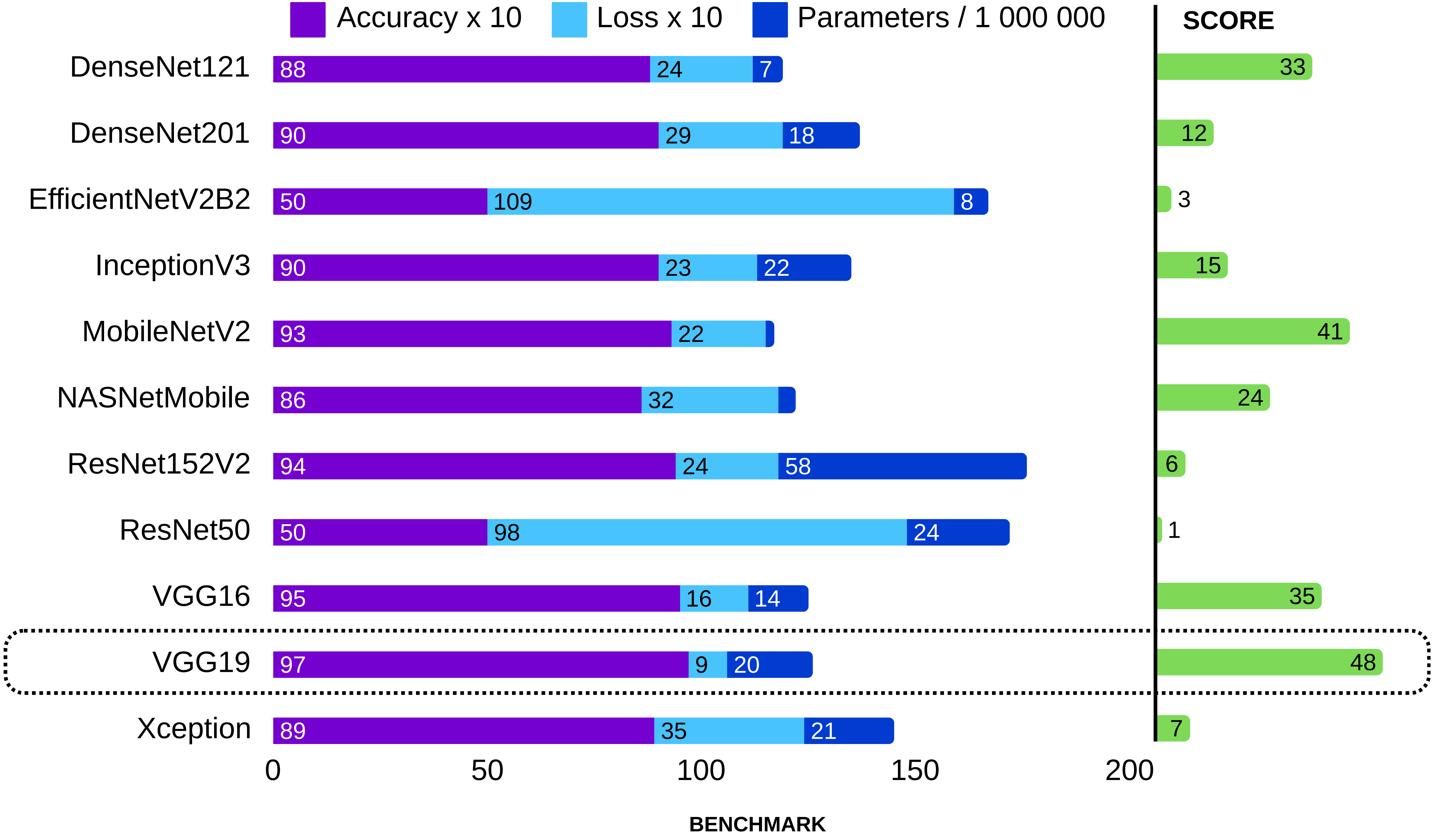
This score favors small models (few parameters) with a high accuracy and a low loss. Small models require less computational power and memory, making them faster to train and deploy.
C *is a constant to adjust the score.*

Here is the list of pre-trained models I used for the benchmark:

- **VGG16**
- **ResNet50**
- **VGG19**
- **ResNet152V2**
- **InceptionV3**
- **InceptionResNetV2**
- **Xception**
- **MobileNetV2**
- **MobileNetV3Large**
- **DenseNet121**
- **DenseNet201**
- **EfficientNetV2B2**
- **NASNetMobile**



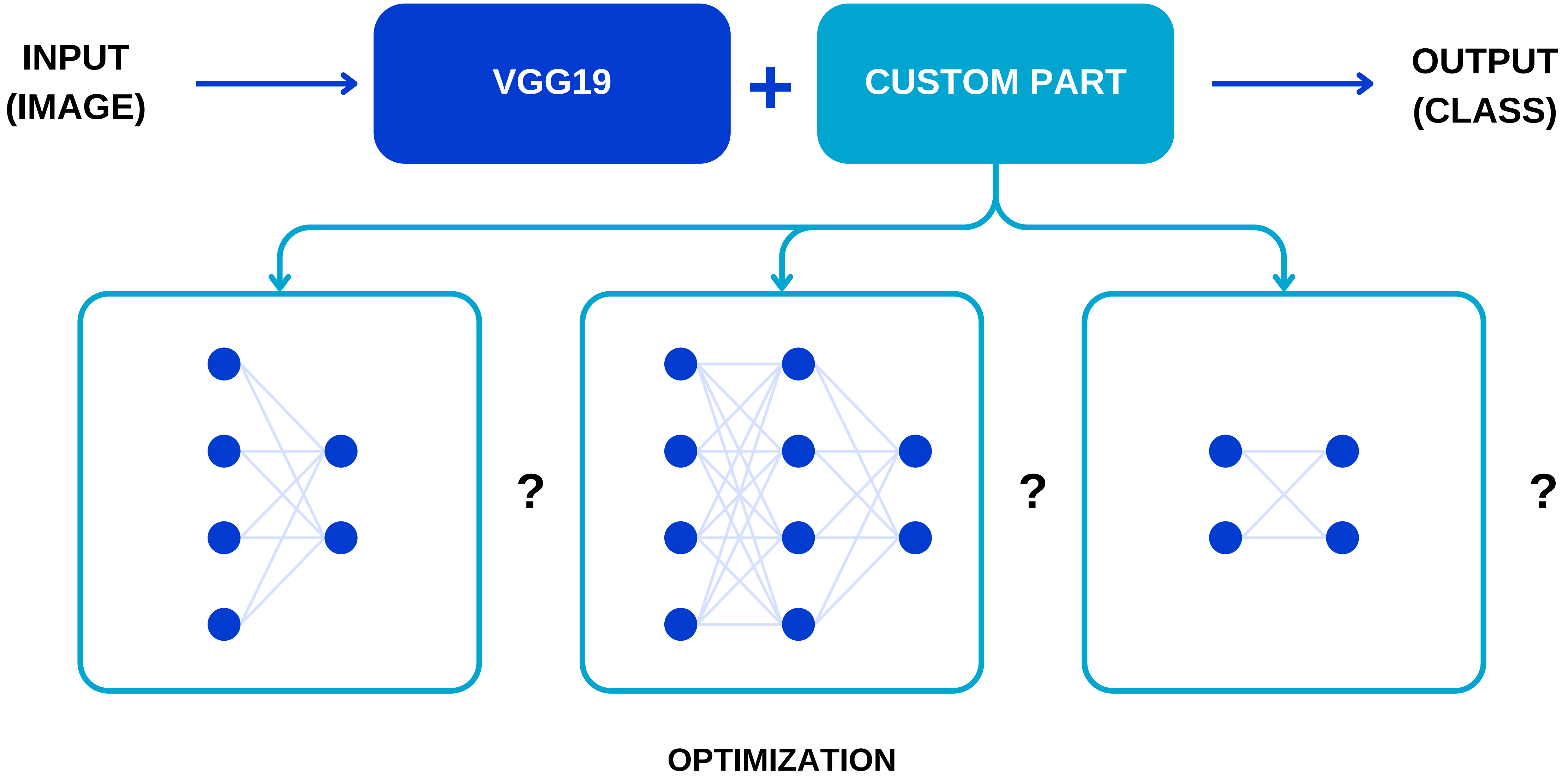
BENCHMARK


Accuracy x 10
Loss x 10
Parameters / 1 000 000
SCORE
DenseNet121 88 24 7 33
DenseNet201 90 29 18 12
EfficientNetV2B2 50 109 8 3
InceptionV3 90 23 22 15
MobileNetV2 93 22 41
NASNetMobile 86 32 24
ResNet152V2 94 24 58 6
ResNet50 50 98 24 1
VGG16 95 16 14 35
VGG19 97 9 20 48
Xception 89 35 21 7
0
50
100
150
200
BENCHMARK

After selecting the pre-trained model (VGG19), I experimented with various architectures (custom part) to identify the most effective configuration for my project.

This involved modifying layers, adjusting hyperparameters, and testing different model designs to optimize accuracy and efficiency.

By systematically evaluating each architecture, I was able to tailor the model to meet the specific requirements of the task while ensuring robust and reliable performance.



Optimization conclusion: **breadth better than depth**

I developed and trained a model based on the VGG19 architecture.

The model was trained on a dataset consisting of **1350** labeled images evenly distributed across the **3 classes**: normal, COVID, and viral pneumonia.

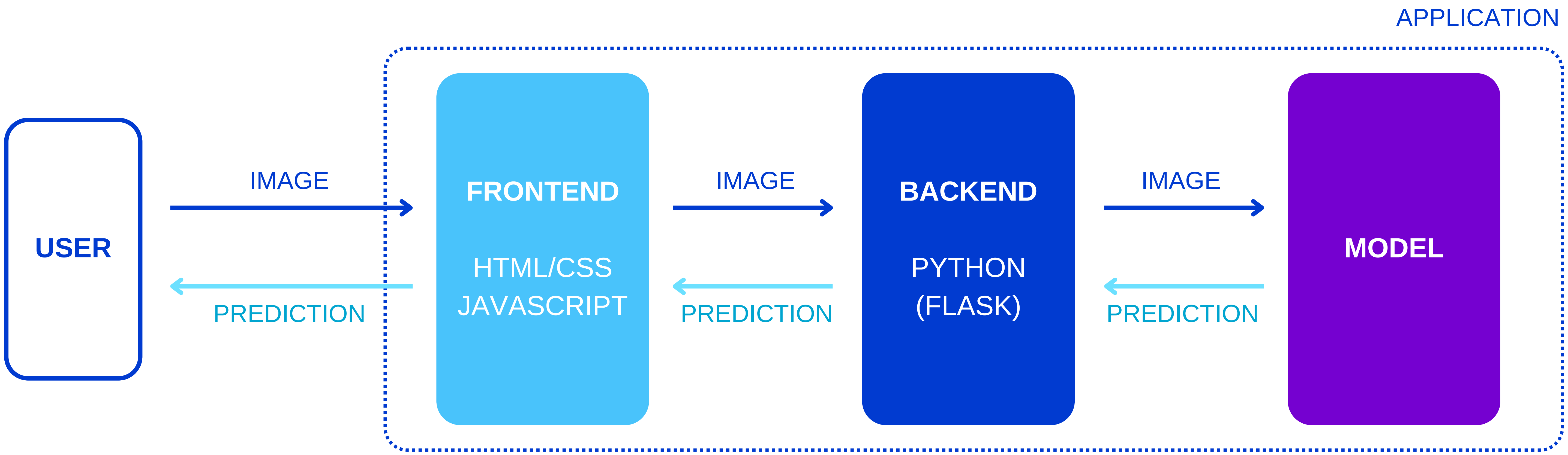
Images were resized to **200x200** pixels, and data augmentation techniques such as rotation, flipping, and brightness adjustment were applied to improve generalization. The training process was conducted over 20 epochs with a **batch size of 64** using the **Adam** optimizer and an initial learning rate of **0.001**.

To monitor performance and avoid overfitting, **early stopping** and **dropout** layers were employed.

After training, the model achieved an accuracy of **98.51%**. This means that out of 100 predictions, the model correctly identifies the class (normal, COVID, or viral pneumonia) approximately 98 times.

BUILD MODEL

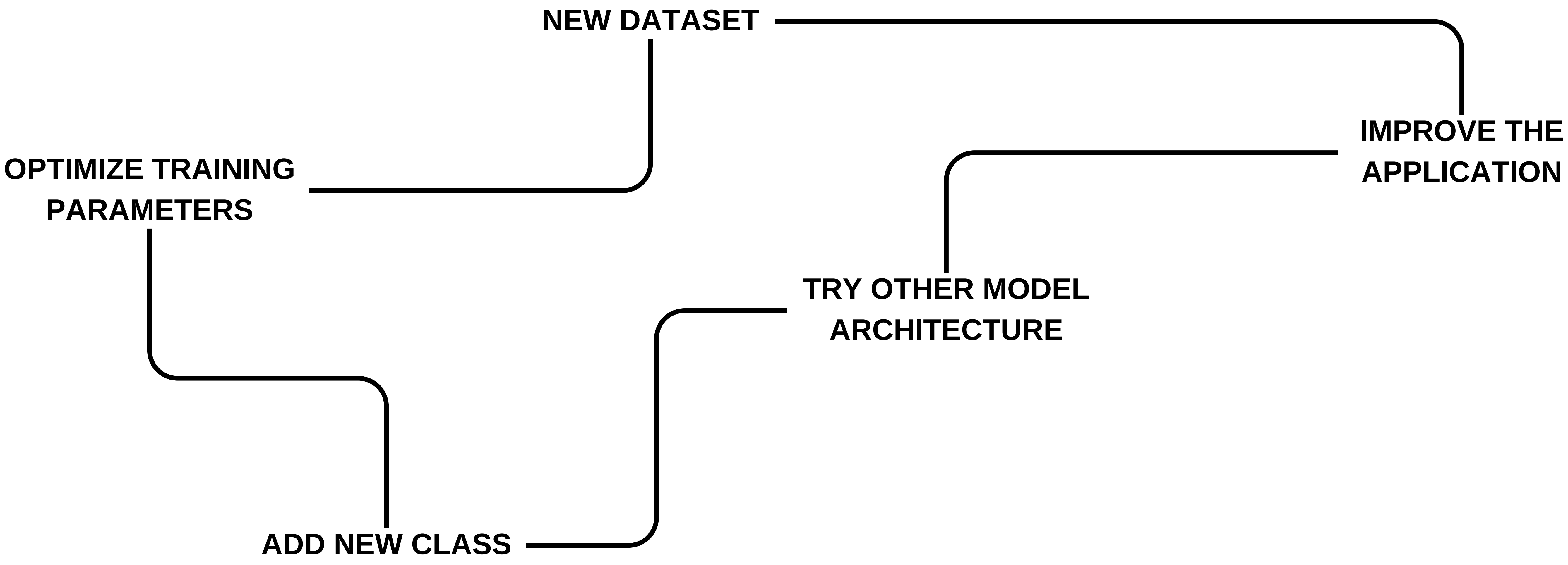
I created a GUI application to interface the user with the model. The user uploads an image, then the application classifles the image using the CNN model.



APPLICATION

This project demonstrates the potential of computer vision when combined with modern machine learning techniques and efficient architectures. While the results are promising, it's essential to remember that this is an Artificial Narrow Intelligence (ANI), specialized for a specific task and lacking the broader understanding or adaptability of human intelligence. Future iterations can build on this foundation, exploring new possibilities and further refining the application.

For the future:



NEW DATASET

IMPROVE THE APPLICATION

OPTIMIZE TRAINING PARAMETERS

TRY OTHER MODEL ARCHITECTURE

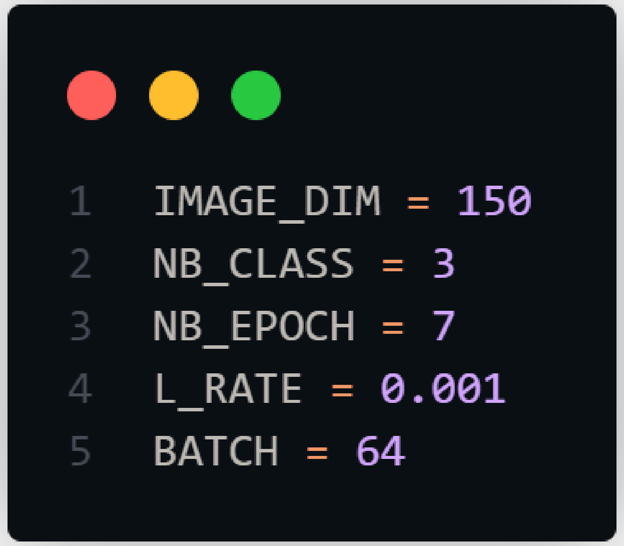
ADD NEW CLASS

CONCLUSION

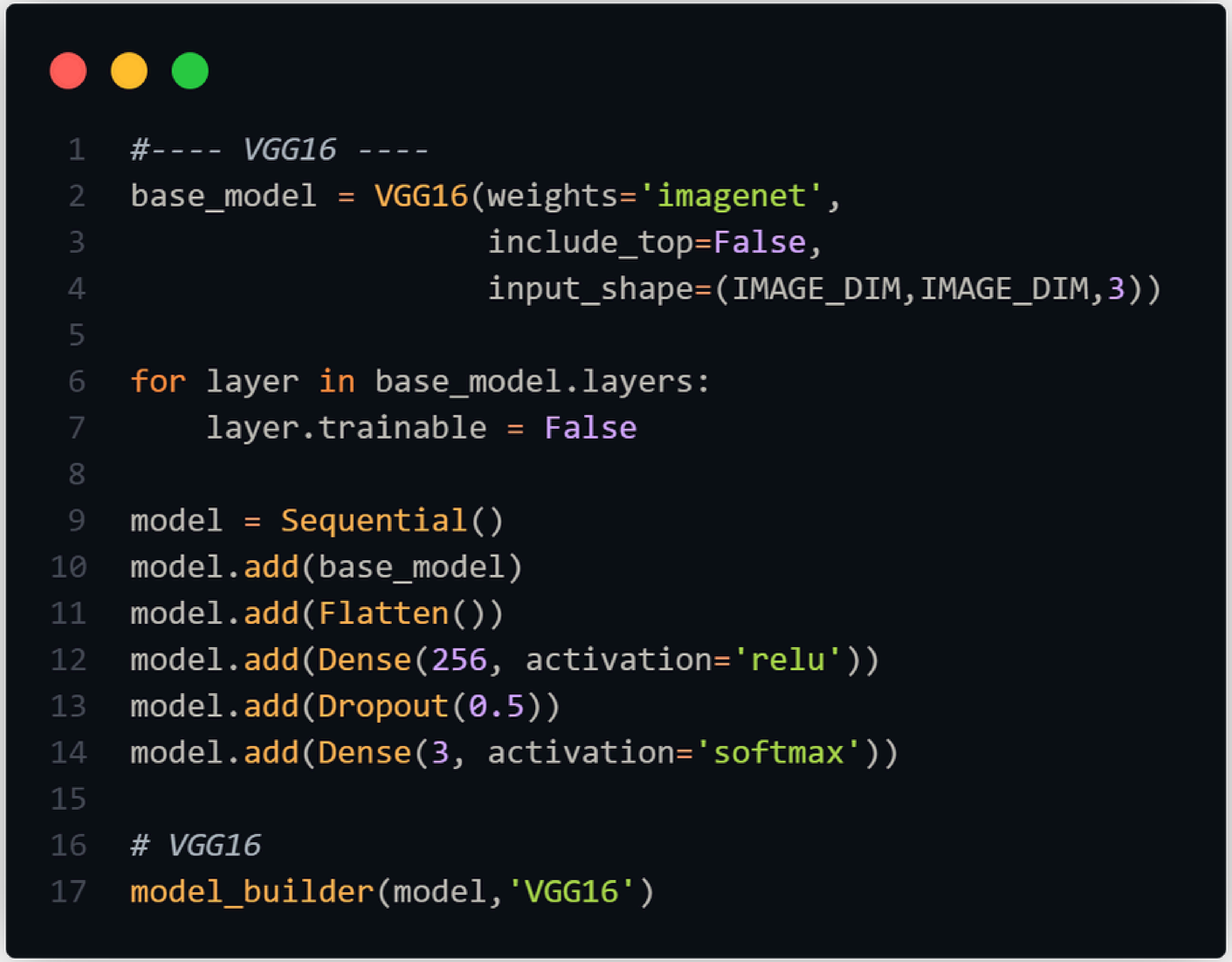
THANK YOU

CONCLUSION

Hyperparameters and model architecture for the benchmark:



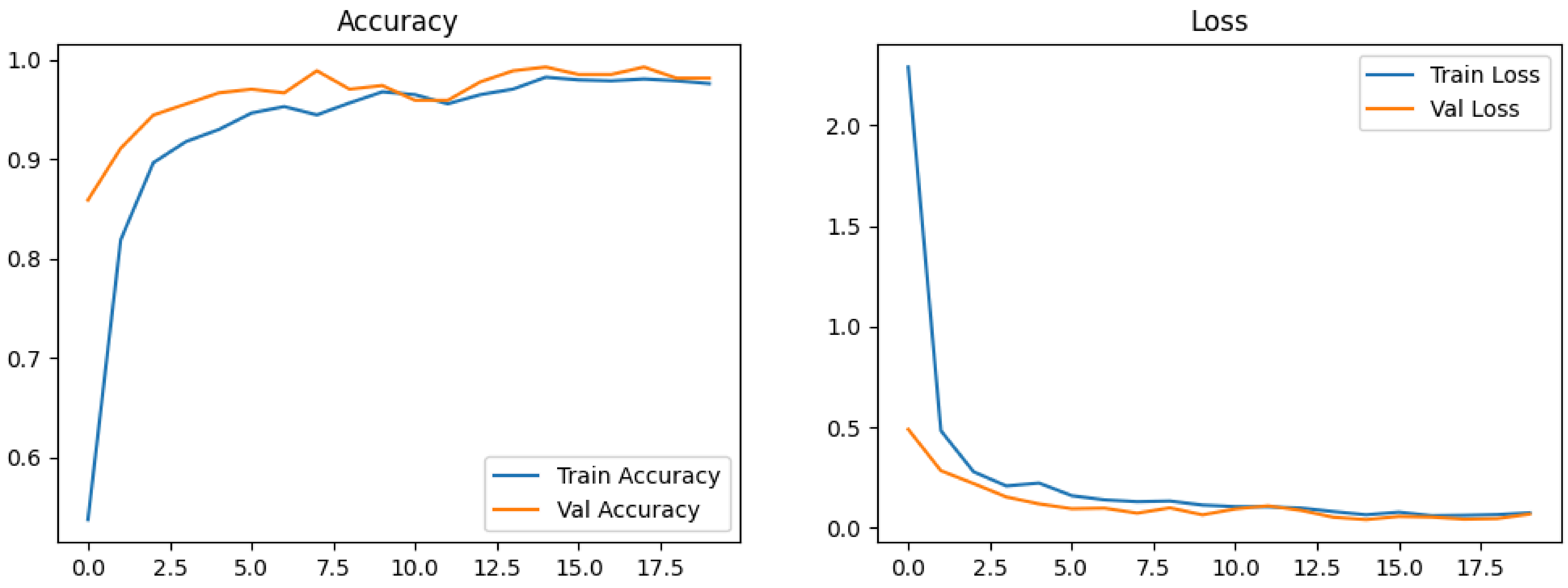
```
IMAGE_DIM = 150
NB_CLASS = 3
NB_EPOCH = 7
L_RATE = 0.001
BATCH = 64
```



```
#---- VGG16 ----
base_model = VGG16(weights='imagenet',
                   include_top=False,
                   input_shape=(IMAGE_DIM,IMAGE_DIM,3))

for layer in base_model.layers:
    layer.trainable = False

model = Sequential()
model.add(base_model)
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax'))

# VGG16
model_builder(model,'VGG16')
```

ADDITIONAL

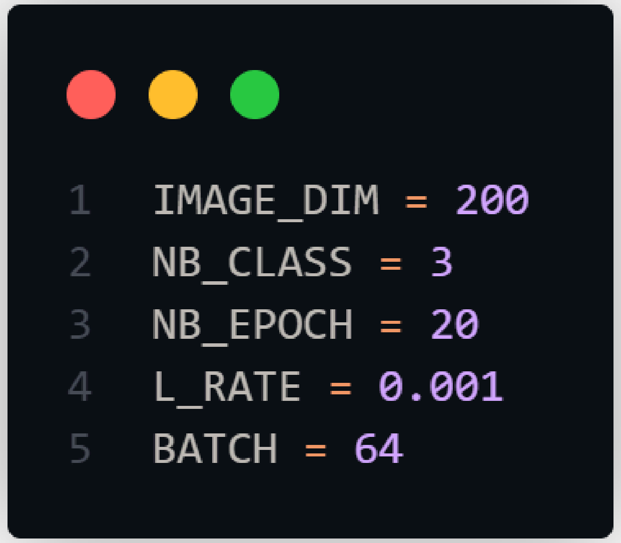
Final model training metrics:



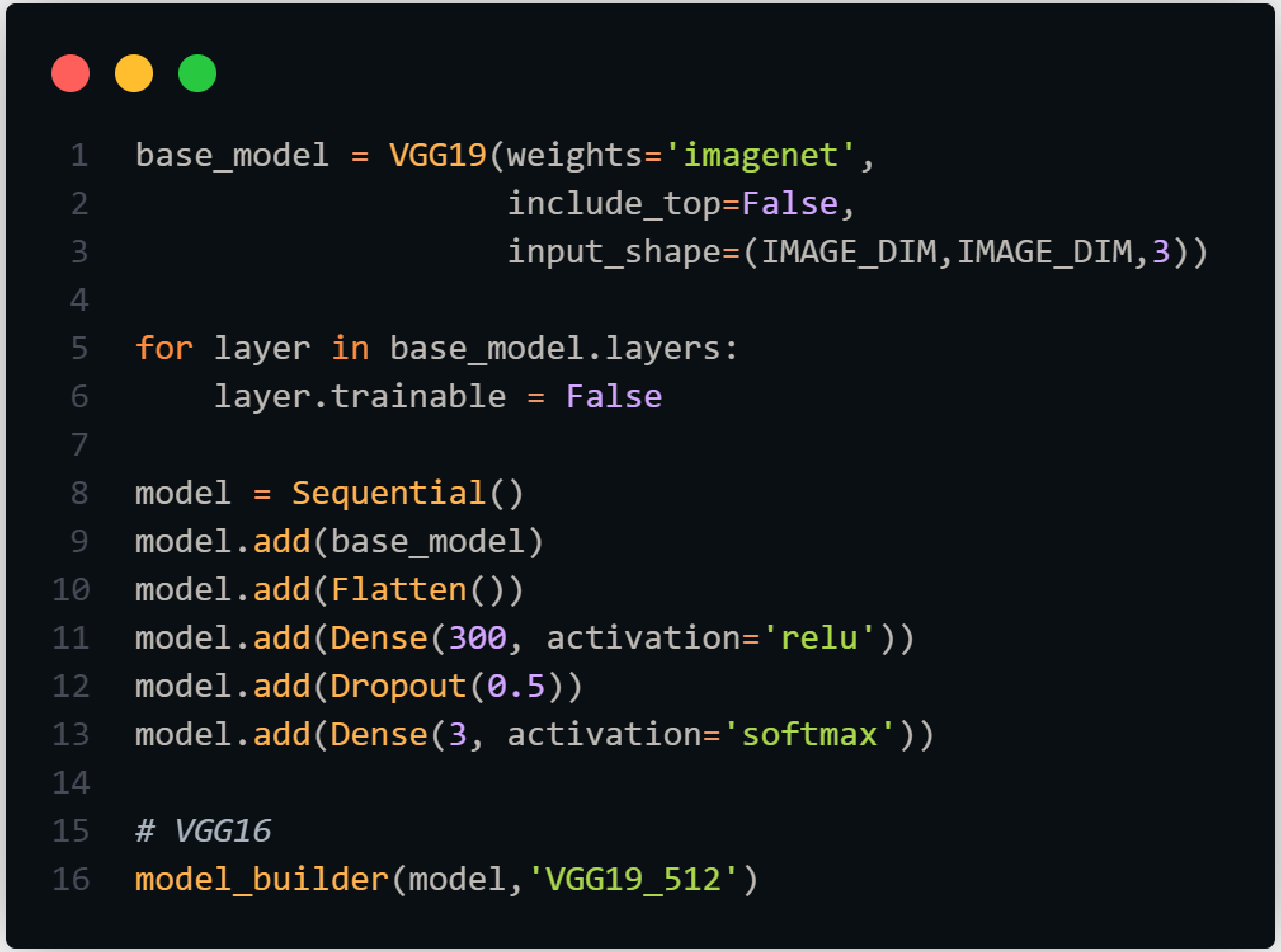
TEST ACCURACY: **98.51%**

ADDITIONAL

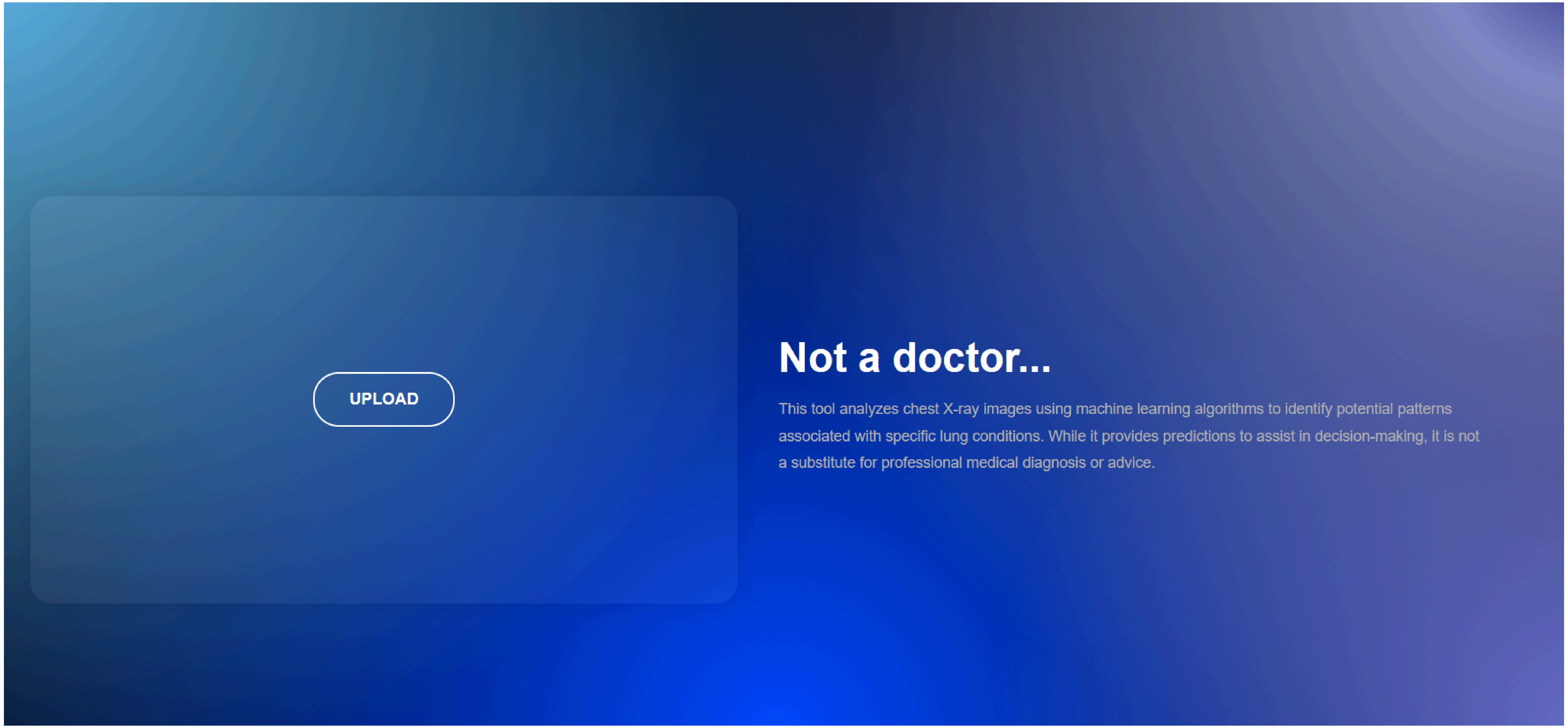
Hyperparameters and model architecture for the main model training:



```
IMAGE_DIM = 200
NB_CLASS = 3
NB_EPOCH = 20
L_RATE = 0.001
BATCH = 64
```



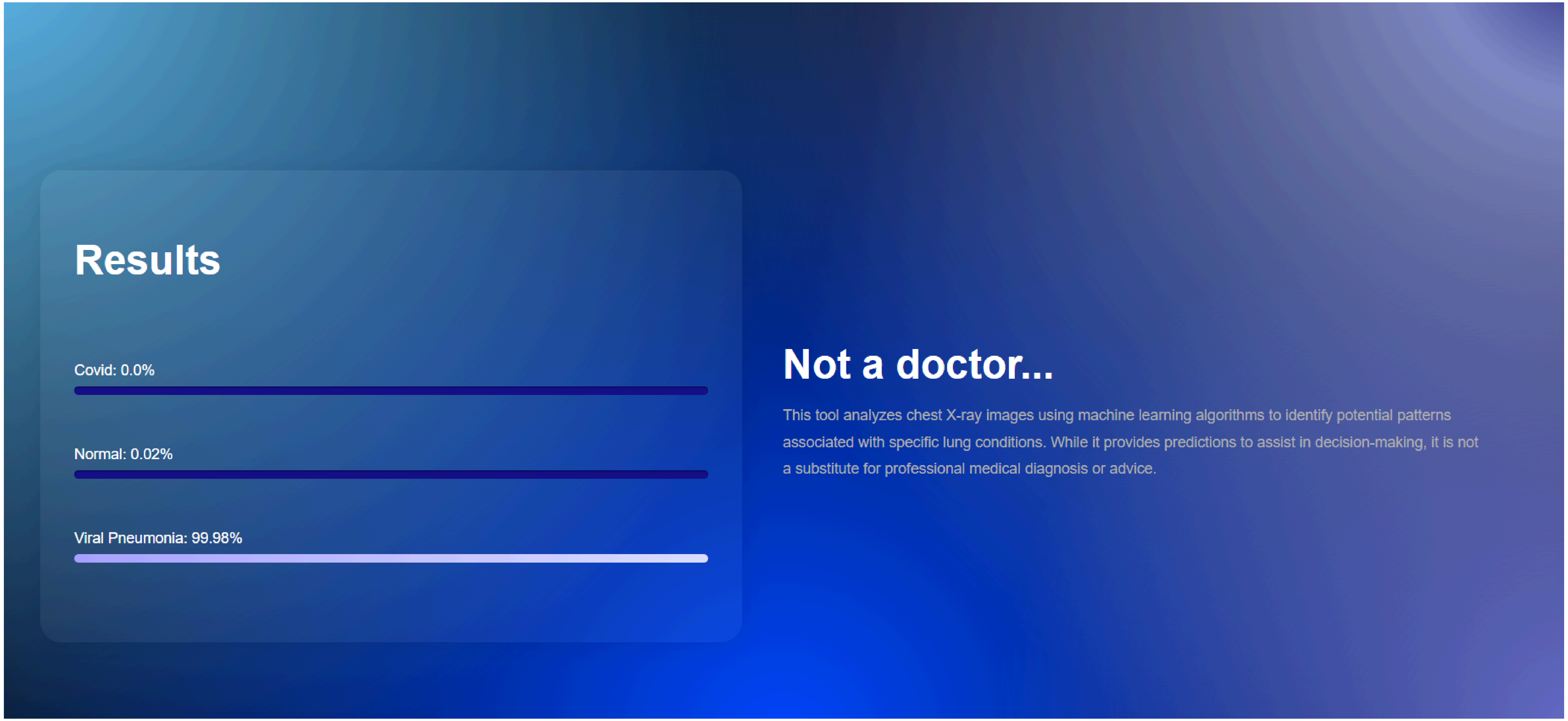
```
base_model = VGG19(weights='imagenet',
                   include_top=False,
                   input_shape=(IMAGE_DIM,IMAGE_DIM,3))

for layer in base_model.layers:
    layer.trainable = False

model = Sequential()
model.add(base_model)
model.add(Flatten())
model.add(Dense(300, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax'))

# VGG16
model_builder(model,'VGG19_512')
```



Not a doctor...

This tool analyzes chest X-ray images using machine learning algorithms to identify potential patterns associated with specific lung conditions. While it provides predictions to assist in decision-making, it is not a substitute for professional medical diagnosis or advice.

SCREENSHOTS



Results

Covid: 0.0%

Normal: 0.02%

Viral Pneumonia: 99.98%

Not a doctor...

This tool analyzes chest X-ray images using machine learning algorithms to identify potential patterns associated with specific lung conditions. While it provides predictions to assist in decision-making, it is not a substitute for professional medical diagnosis or advice.

SCREENSHOTS