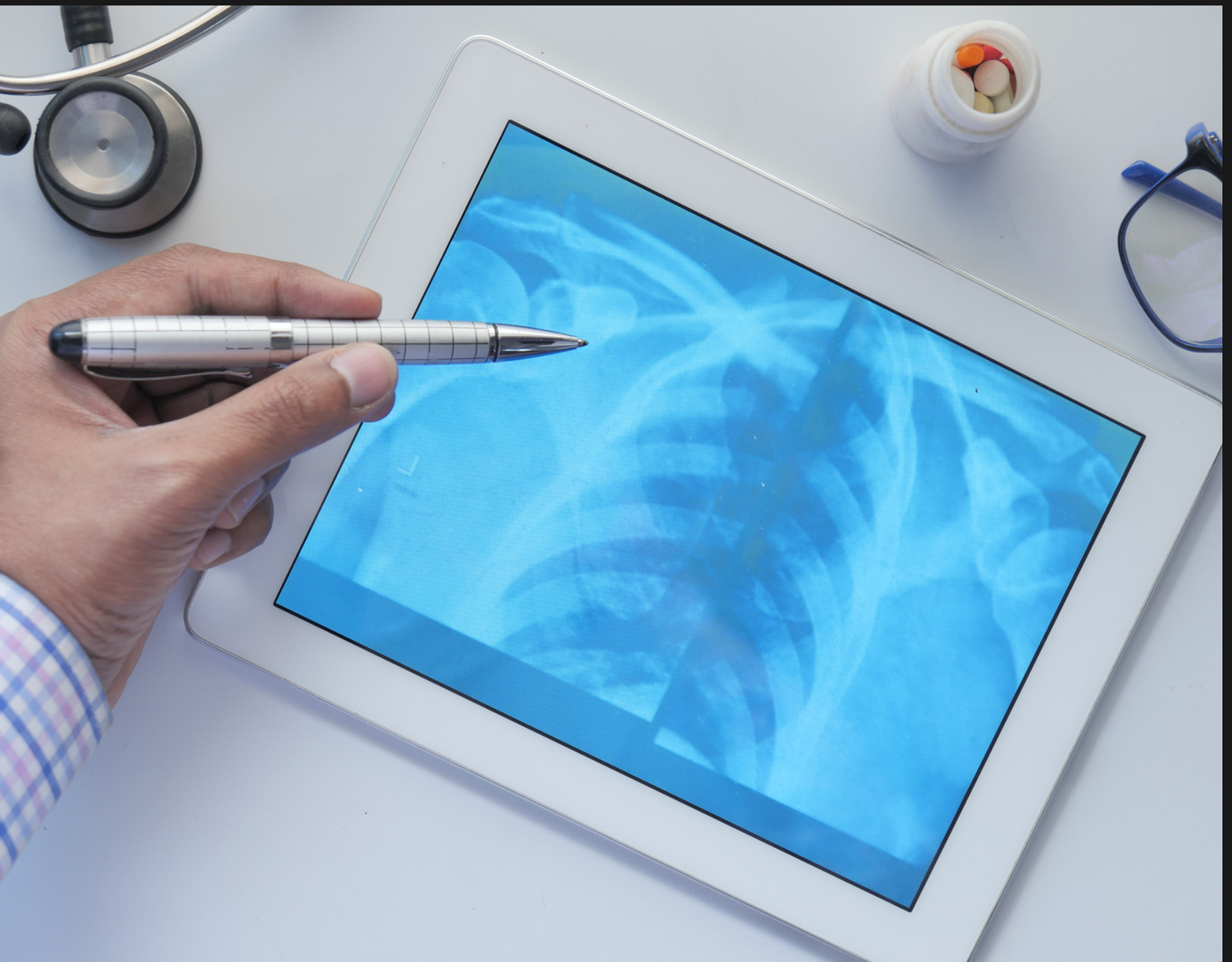


Classification on Medical Image

Case Study Industry

AI in Healthcare

AI in Healthcare

The computing capability of deep learning models has enabled fast, accurate and efficient operations in healthcare. Deep learning networks are transforming patient care and they have a fundamental role for health systems in clinical practice. Computer vision, natural language processing, reinforcement learning are the most commonly used deep learning techniques in healthcare.

AI Usecases in Healthcare

Patient Care
- Medical imaging
- Healthcare data analytics
- Mental health chatbots
- Personalized medical treatments
- Prescription audit
- Responding to patient queries

Health Insurance
- Underwriting
- Fraud detection

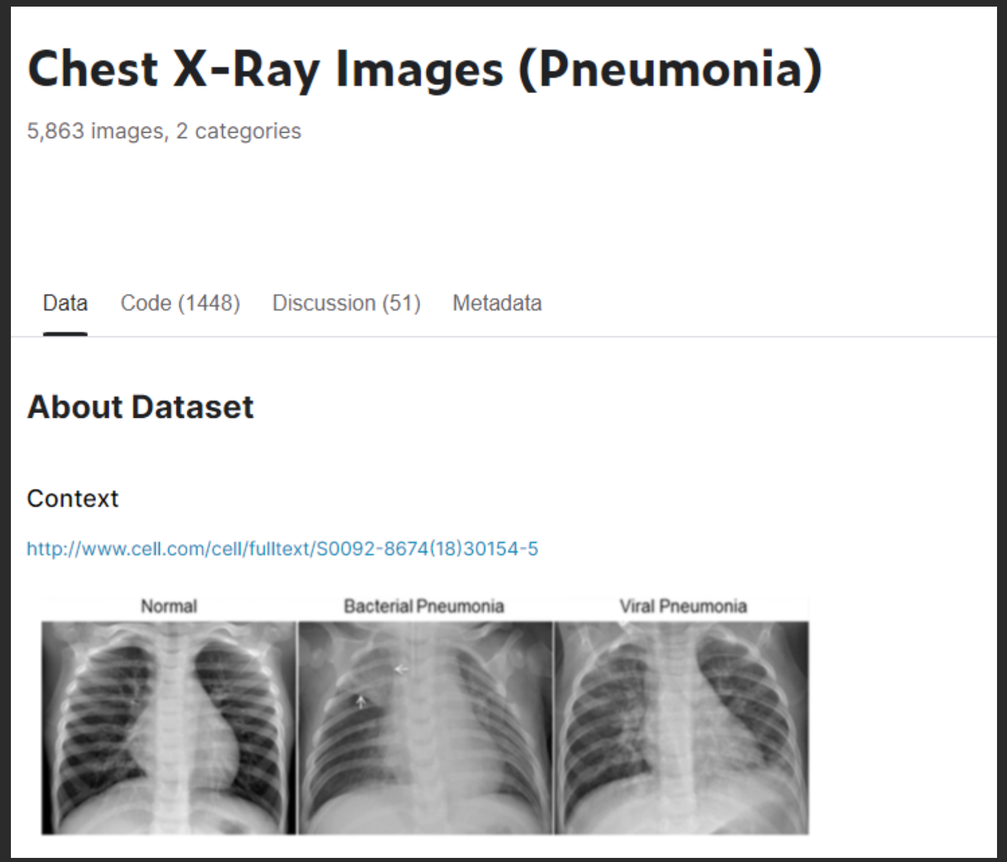
Research & Development
- Drug discovery
- Genomics analysis
- Mental health research
- Covid-19

ref : https://research.aimultiple.com/deep-learning-in-healthcare/

Data Science Method for Medical Image Usecase

Medical Image Dataset

Kaggle Dataset : https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia



The dataset is organized into 3 folders (train, test, val) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) and 2 categories (Pneumonia/Normal).

Chest X-ray images (anterior-posterior) were selected from retrospective cohorts of pediatric patients of one to five years old from Guangzhou Women and Children's Medical Center, Guangzhou. All chest X-ray imaging was performed as part of patients' routine clinical care.

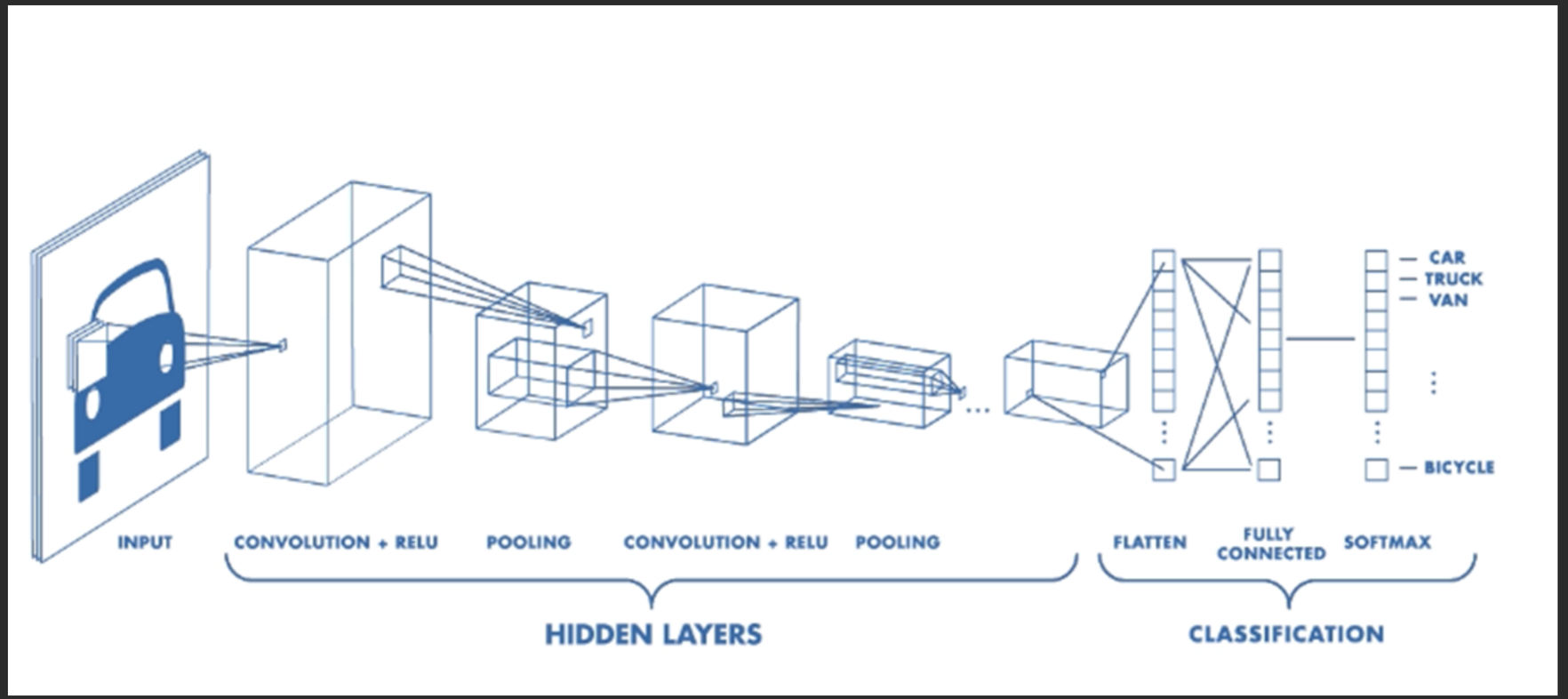
For the analysis of chest x-ray images, all chest radiographs were initially screened for quality control by removing all low quality or unreadable scans. The diagnoses for the images were then graded by two expert physicians before being cleared for training the AI system. In order to account for any grading errors, the evaluation set was also checked by a third expert.

Medical Image Dataset

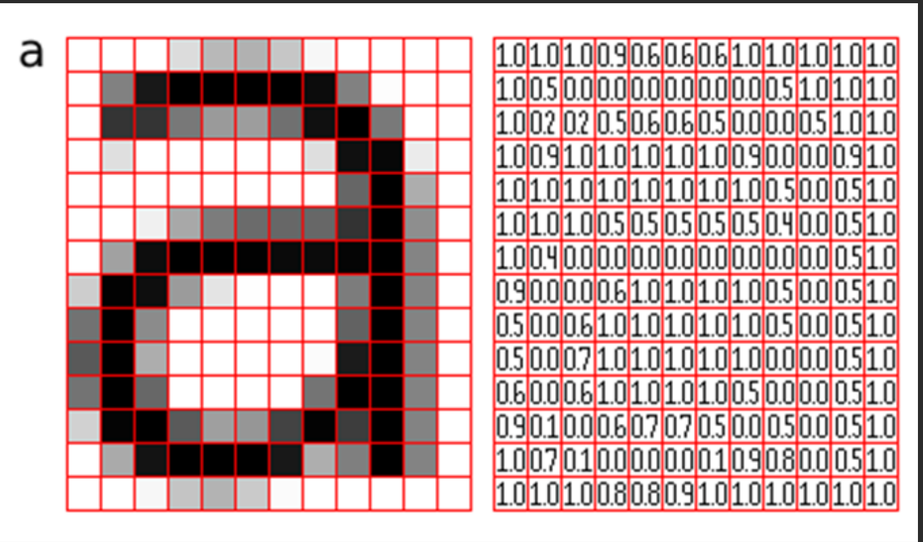
Illustrative Examples of Chest X-Rays in Patients with Pneumonia

The normal chest X-ray (left panel) depicts clear lungs without any areas of abnormal opacification in the image. Bacterial pneumonia (middle) typically exhibits a focal lobar consolidation, in this case in the right upper lobe (white arrows), whereas viral pneumonia (right) manifests with a more diffuse “interstitial” pattern in both lungs.

Convolutional Neural Network (CNN)



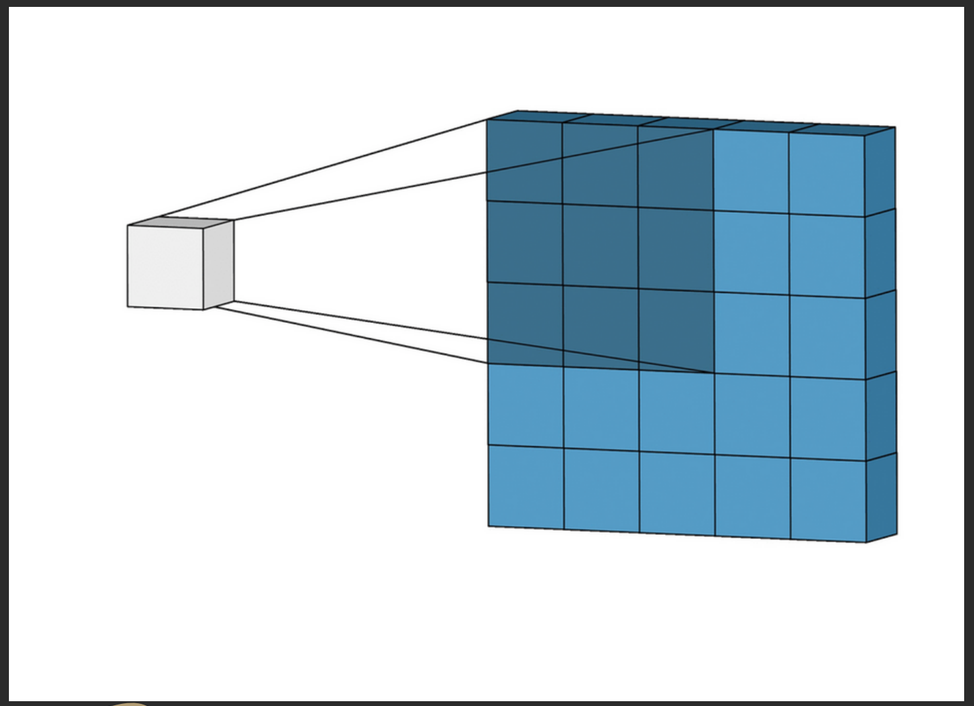
CNN (cont'd)



Convolutional Neural Network

A Convolutional Neural Network, also known as CNN or ConvNet, is a class of neural networks that specializes in processing data that has a grid-like topology, such as an image. A digital image is a binary representation of visual data. It contains a series of pixels arranged in a grid-like fashion that contains pixel values to denote how bright and what color each pixel should be.

CNN (cont'd)

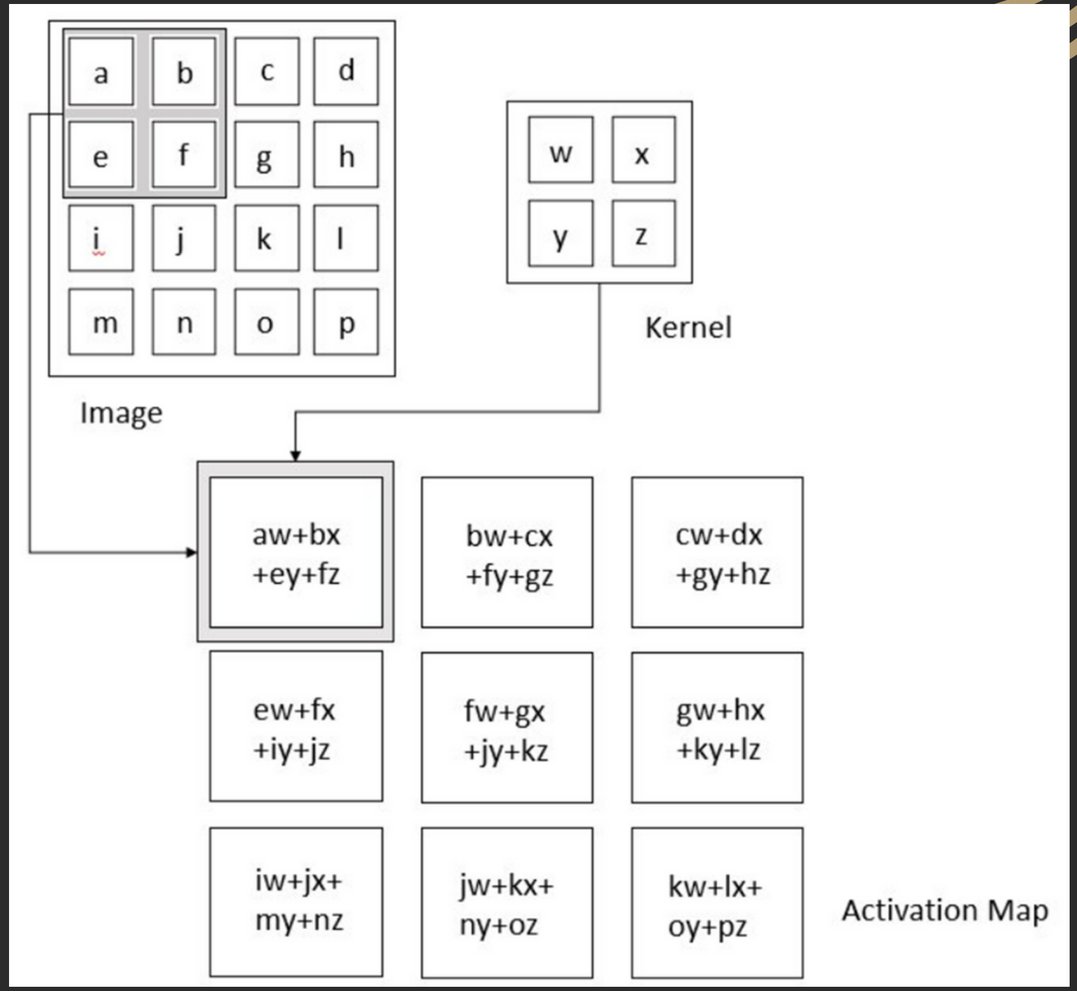


Convolution Layer

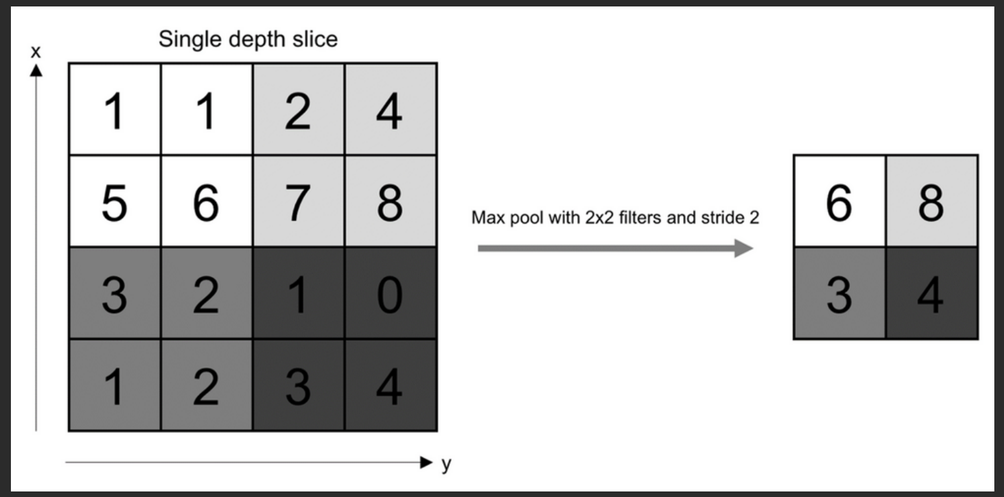
The convolution layer is the core building block of the CNN. It carries the main portion of the network's computational load.

This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels.

CNN (cont'd)


a b c d
e f g h
i j k l
m n o p
Image
w x
y z
Kernel
aw+bx
+ey+fz
bw+cx
+fy+gz
cw+dx
+gy+hz
ew+fx
+iy+jz
fw+gx
+jy+kz
gw+hx
+ky+lz
iw+jx+
my+nz
jw+kx+
ny+oz
kw+lx+
oy+pz
Activation Map

CNN (cont'd)



Pooling Layer

The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

There are several pooling functions such as the average of the rectangular neighborhood, L2 norm of the rectangular neighborhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighborhood.

CNN (cont'd)

Non-Linearity Layers

Since convolution is a linear operation and images are far from linear, non-linearity layers are often placed directly after the convolutional layer to introduce non-linearity to the activation map. There are several types of non-linear operations, the popular ones being: Sigmoid, Tanh, ReLU, etc.

Fully Connected Layer

Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular FCNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect. The FC layer helps to map the representation between the input and the output.

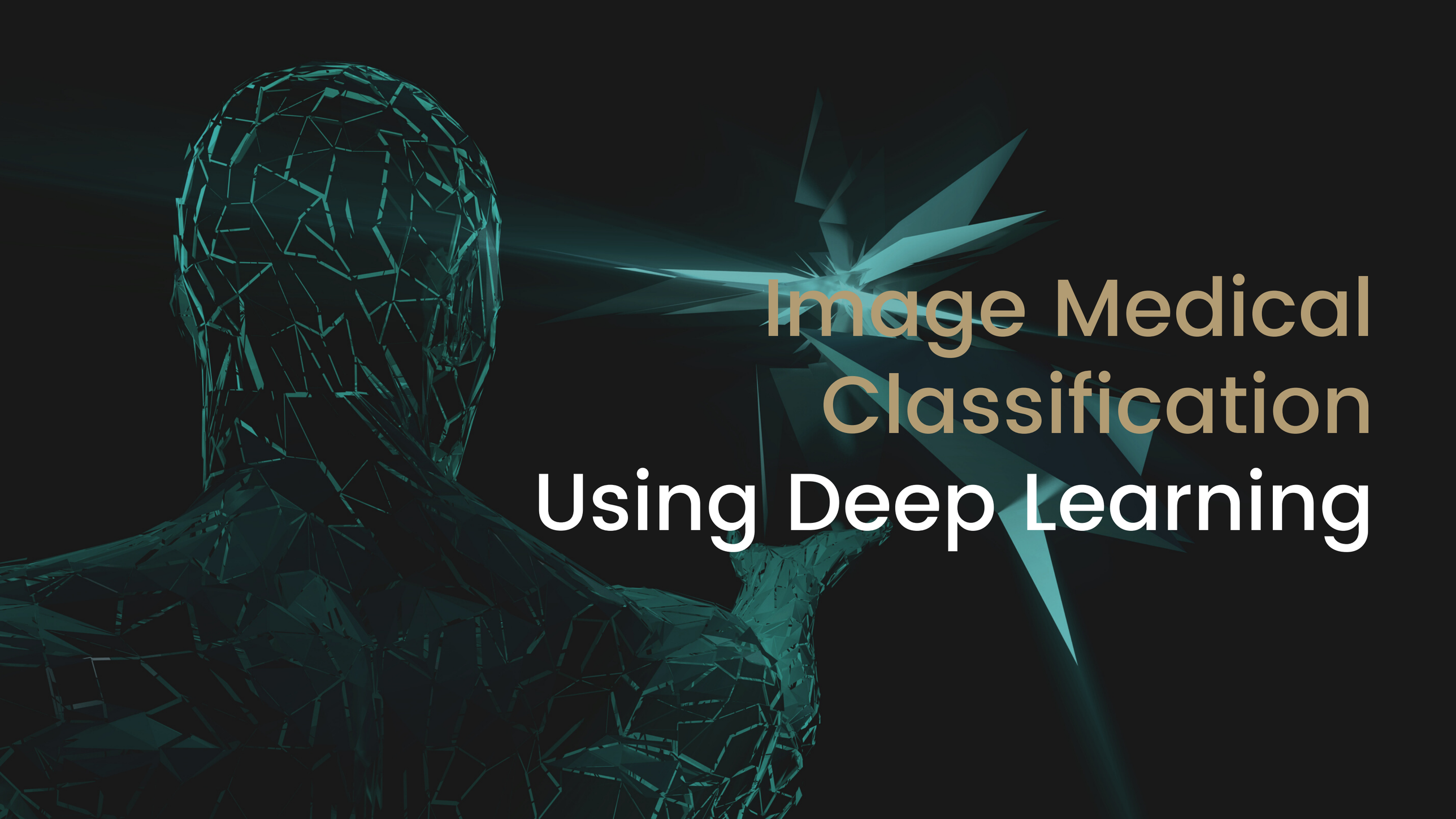


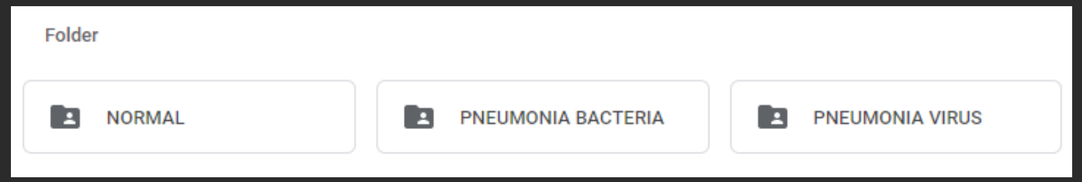
Image Medical Classification Using Deep Learning

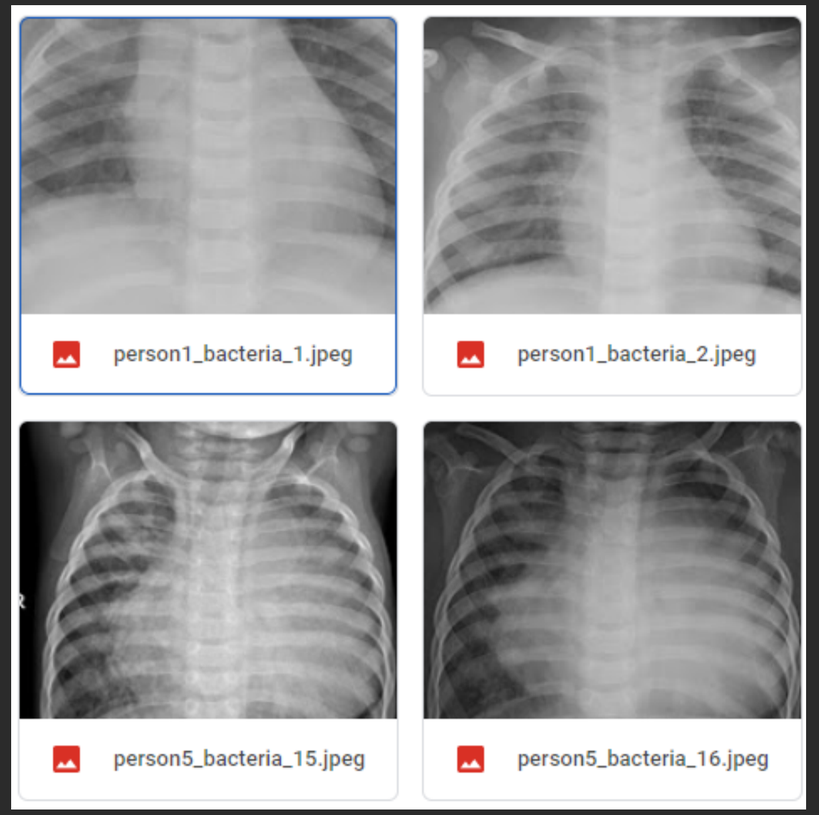
Preparing Dataset

The dataset is organized into 3 folders based on label/ target. Python script will read these 3 folder and setup label based on folder name. In this step, we use OpenCV library to read images.

```
pics, labels = [], []
for dir_path in glob.glob("/content/drive/MyDrive/MSIB/15 - Case Study Industri (Klasifikasi Gambar)/train/*"):
  for pic in glob.glob(os.path.join(dir_path,"*.jpeg")):
    temp = cv2.imread(pic)
    temp = cv2.resize(temp, (70, 70))
    pics.append(temp)
    word_label = os.path.basename(os.path.dirname(pic))
    labels.append(word_label)

x_train = np.array(pics)
y_train = np.array(labels)
```





Data Preprocessing

Image Reshape/ Resize

```
temp = cv2.imread(pic)
temp = cv2.resize(temp, (70, 70))
```

Label Encoding

```
from sklearn import preprocessing

le = preprocessing.LabelEncoder()
le.fit(y_train)
y_train_le = le.transform(y_train)
y_test_le = le.transform(y_test)
```

MinMax Scaling

```
# PREPROCESSING
X_train = x_train.astype('float32') #set x_train data type as float32
X_test = x_test.astype('float32') #set x_test data type as float32
X_train /= 255 #change x_train value between 0 - 1
X_test /= 255 #change x_test value between 0 - 1
```

One Hot Encoding

```
from tensorflow.keras.utils import to_categorical
y_train_ohe = to_categorical(y_train_le,3)
y_test_ohe = to_categorical(y_test_le,3)
```

Define Model

```
# ARSITEKTUR
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Dropout, Flatten, Dense
model = Sequential() #model = sequential
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',input_shape=(70,70,3))) #layer convolutional 2D
model.add(MaxPooling2D(pool_size=(2,2))) #max pooling with stride (2,2)
model.add(Conv2D(32, (3, 3), activation='relu')) #layer convolutional 2D
model.add(MaxPooling2D(pool_size=(2,2))) #max pooling with stride (2,2)
model.add(Dropout(0.25)) #delete neuron randomly while training and remain 75%
model.add(Flatten()) #make layer flatten
model.add(Dense(128, activation='relu')) #fully connected layer
model.add(Dropout(0.5)) #delete neuron randomly and remain 50%
model.add(Dense(3, activation='softmax')) #softmax works
```

define architecture model

```
# COMPILE
from tensorflow.keras.optimizers import SGD
epochs = 50
lrate = 0.01
decay = lrate/epochs
sgd = SGD(lr=lrate, momentum=0.9, decay=decay, nesterov=False)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
print(model.summary())
```

```
from keras.callbacks import ModelCheckpoint
filepath=r"/content/drive/MyDrive/MSIB/15 - Case Study Industri (Klasifikasi Gambar)/model3classes.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True, mode='max')
```

define saving model criteria

```
#TRAINING
model.fit(X_train, y_train_ohe, validation_data=(X_test, y_test_ohe), epochs=epochs, batch_size=32, callbacks=[checkpoint])
scores = model.evaluate(X_test, y_test_ohe, verbose=0)
```

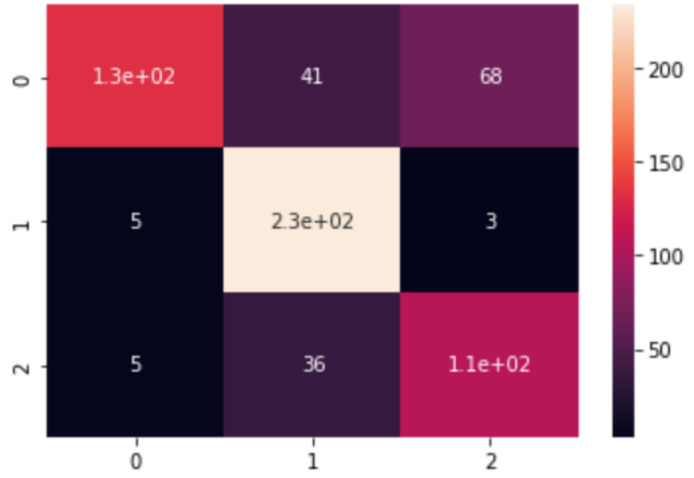
Model Evaluation

```
#CLASSIFICATION REPORT
from sklearn.metrics import classification_report
print(classification_report(y_hasil, y_test))
```

	precision	recall	f1-score	support
NORMAL	0.55	0.93	0.69	143
PNEUMONIA BACTERIA	0.97	0.75	0.85	311
PNEUMONIA VIRUS	0.72	0.60	0.66	178
accuracy			0.75	632
macro avg	0.75	0.76	0.73	632
weighted avg	0.80	0.75	0.76	632

```
from sklearn.metrics import confusion_matrix, accuracy_score
import seaborn as sn
print('accuracy score : ',accuracy_score(y_test, y_hasil))
labels =['NORMAL','PNEUMONIA BACTERIA','PNEUMONIA VIRUS']
cm = confusion_matrix(y_test, y_hasil, labels=labels)
sn.heatmap(cm, annot=True)
```

```
accuracy score :  0.75
<matplotlib.axes._subplots.AxesSubplot at 0x7fc17c11c790>
```



References

- Deep Learning by Ian Goodfellow, Yoshua Bengio and Aaron Courville published by MIT Press, 2016
- Stanford University’s Course — CS231n: Convolutional Neural Network for Visual Recognition by Prof. Fei-Fei Li, Justin Johnson, Serena Yeung
- https://datascience.stackexchange.com/questions/14349/difference-of-activation-functions-in-neural-networks-in-general
- https://www.codementor.io/james_aka_yale/convolutional-neural-networks-the-biologically-inspired-model-iq6s48zms
- https://searchenterpriseai.techtarget.com/definition/convolutional-neural-network
- https://docs.opencv.org/4.x/d2/d96/tutorial_py_table_of_contents_imgproc.html


Thank You