UEC-704

Soft Computing

Classification of chest x-ray samples to detect COVID-19 using Convolutional Neural Networks

Thapar Institute of Engineering & Technology

Submitted by:			Submitted to:
1 Param Prashar	101706112	L2	Dr Gaganpreet
2 Romit Taneja	101765004	L2	
3 Saksham Mandhyan	101706135	L2	
4 Siddhant Singh Rawat	101715154	ENC3	
5 Sparsh Goel	101765007	ENC3	

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Objective:

Classification of chest x-ray samples to detect COVID-19 using Convolutional Neural Networks

In early 2020, a new virus began making headlines all over the world, because of the unprecedented speed of its transmission. Its origins have been traced to a food market in Wuhan, China, in December 2019. From there, it's reached countries as distant as the United States and the Philippines. The virus (officially named SARS-CoV-2) has been responsible for millions of infections globally, causing hundreds of thousands of deaths. India remains the country most affected. The disease caused by an infection with SARS-CoV-2 is called COVID-19, which stands for coronavirus disease 2019.

This study aims to use X-Ray sample images from lungs of people infected with covid and the un-affected ones as an input to a Convolution Neural Network. This CNN, after training and validation, can then be used as a model to classify new samples into infected and normal lungs. This study brings with it, a huge scope in the field of medical imaging and can be revolutionary in refined formats, to be used worldwide for better understanding and quick testing of the disease.

Recent works:

Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks

Boran Sekeroglu and Ilker Ozsahin

This research paper explains detection of severe acute respiratory syndrome coronavirus 2 (SARS CoV-2), which is responsible for coronavirus disease 2019 (COVID-19), using chest X-ray images and how it has life-saving importance for both patients and doctors. The study done here focuses on the present use of deep learning models, specifically model based on deep convolutional neural networks (ConvNets), for the high-accuracy detection of COVID-19 using chest X-ray images. The dataset taken in the study comprises of the publicly available X-ray images (1583 healthy, 4292 pneumonia, and 225 confirmed COVID-19). This data set was then used in the several experiments, which involved the training of deep learning and machine learning classifiers. Thirty-eight experiments were performed using deep convolutional neural networks, 10 experiments were performed using five machine learning models, and 14 experiments were performed using the state-of-the-art pre-trained networks for transfer learning. Images and statistical data were considered separately in the experiments to evaluate the performances of models, and eightfold cross-validation was used. The training experiments provide a thorough insight on the performance of different machine learning models and various deep convolutional neural networks architecture against the X-ray dataset. A mean sensitivity of 93.84%, mean specificity of 99.18%, mean accuracy of 98.50%, and mean receiver operating characteristics–

area under the curve scores of 96.51% are achieved. The useful drawn conclusion regarding the performance of different models opens a whole new opportunity for the pandemic treatment and research field. In addition, in countries that are unable to purchase laboratory kits for testing, this becomes even more vital. It provides an efficient procedure for the detection of COVID-19 from chest X-ray images which not only decrease the diagnostic time but also reduces financial cost. Further studies, based on the results obtained in this study, would provide more information about the use of CNN architectures with COVID-19 chest X-ray images and improve on the results of this study.

A Combined Deep CNN-LSTM Network for the Detection of Novel Coronavirus (COVID-19) Using X-ray Images

M. Z. Islam, M. M. Islam, A. Asraf

The study proposed in this model talks about automatic disease detection becoming a crucial issue in medical science with the rapid growth of population. Coronavirus (COVID-19) has become one of the most severe and acute diseases in very recent times that has been spread globally. Automatic disease detection framework assists the doctors in the diagnosis of disease and provides exact, consistent, and fast reply as well as reduces the death rate. Therefore, an automated detection system should be implemented as the fastest way of diagnostic option to impede COVID-19 from spreading. This paper aims to introduce a deep learning technique based on the combination of a convolutional neural network (CNN) and long short -term memory (LSTM) to diagnose COVID-19 automatically from X-ray images. In this system, CNN is used for deep feature extraction and LSTM is used for detection using the extracted feature. A collection of 421 X -ray images including 141 images of COVID-19 is used as a dataset in this system. The experimental results show that this paper's proposed system has achieved 97% accuracy, 91% specificity, and 100% sensitivity. The system achieved desired results on a small dataset which can be further improved when more COVID-19 images become available. The proposed system can assist doctors to diagnose and treatment the COVID-19 patients easily.

COVID Faster R-CNN: A Novel Framework to Diagnose Novel Coronavirus Disease (COVID19) in X-Ray Images

Kabid Hassan Shibly , Samrat Kumar Dey, Md. Tahzib-Ul-Islam , and Md. Mahbubur Rahman

This research begins while discussing how COVID-19 or novel coronavirus disease, has already been declared as a worldwide pandemic, at first had an outbreak in a small town of China, named Wuhan. More than two hundred countries around the world have already been affected by this severe virus as it spreads by human interaction. Moreover, the symptoms of novel coronavirus are quite similar to the general flu. Screening of infected patients is considered as a critical step in the fight against COVID-19. Therefore, it is highly relevant to recognize positive cases as early as possible to avoid further spreading of this epidemic. However, there are several methods to detect COVID-19 positive patients, which are typically performed based on respiratory samples and among them one of the critical approaches which is treated as radiology imaging or X-Ray imaging. Recent findings from X-Ray imaging techniques suggest that such images contain relevant information about the SARS-CoV-2 virus. In this article, we have introduced a Deep Neural Network (DNN) based Faster Regions with Convolutional Neural Networks (Faster R-CNN) framework to detect COVID-19 patients from chest X-Ray images using available open-source dataset. The proposed model and approach provides a classification accuracy of 97.36%, 97.65% of sensitivity, and a precision of 99.28%. Therefore, we believe this proposed method might be of assistance for health professionals to validate their initial assessment towards COVID-19 patients.

Pneumonia Detection Using CNN based Feature Extraction

Dimpy Varshni, K. Thakral, Lucky Agarwal, Rahul Nijhawan, A. Mittal

This paper, written in earlier times shows how pneumonia a life-threatening infectious disease affecting one or both lungs in humans commonly caused by bacteria called Streptococcus pneumoniae, can be also detected using Xray imaging and machine learning. One in three deaths in India is caused due to pneumonia as reported by World Health Organization (WHO). Chest X-Rays which are used to diagnose pneumonia need expert radiotherapists for evaluation. Thus, developing an automatic system for detecting pneumonia would be beneficial for treating the disease without any delay particularly in remote areas. Due to the success of deep learning algorithms in analyzing medical images, Convolutional Neural Networks (CNNs) have gained much attention for disease classification. In addition, features learned by pre-trained CNN models on large-scale datasets are much useful in image classification tasks. In this work, we appraise the functionality of pre-trained CNN models utilized as feature-extractors followed by different classifiers for the classification of abnormal and normal chest X-Rays. We analytically determine the optimal CNN model for the purpose. Statistical results obtained demonstrates that pretrained CNN models employed along with supervised classifier algorithms can be very beneficial in analysing chest X-ray images, specifically to detect Pneumonia.

Pneumonia Diagnosis Using Chest X-ray Images and Machine Learning

Sara Lee Kit Yee, W. Raymond

In this paper, a pneumonia diagnosis system was developed using convolutional neural network (CNN) based feature extraction. InceptionV3 CNN was used to perform feature extraction from chest X-ray images. The extracted feature was used to train three classification algorithm models to predict the cases of pneumonia from a Kaggle dataset. The three models are K-Nearest Neighbor, Neural Network, and Support Vector Machines. Performance evaluation and confusion matrix were presented to represent the sensitivity, accuracy, precision, and specificity of each of the models. Results show that the Neural Network model achieved the highest sensitivity of 84.1%, followed by Support vector machines (83.5%) and K-Nearest Neighbor Algorithm (83.3%). Among all the classification models, Support vector machines model achieved the highest AUC of 93.1%.

Block Diagram:

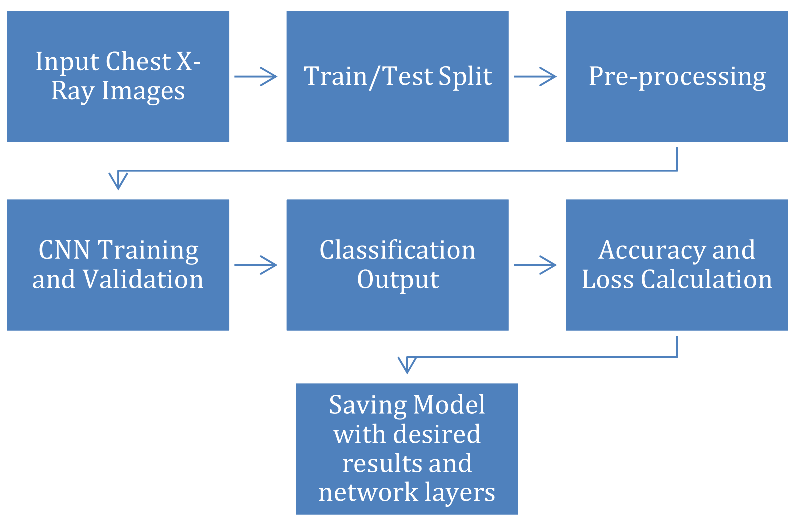

Figure 1: Block diagram of process flow of the proposed CNN model.

A total of 130 available chest x-ray samples are used to train the concurrent neural network. Tensorflow keras is used in implementation of the said model. In preprocessing images are recaled to the size of 64x 64.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 62, 62, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 31, 31, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 29, 29, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 6272)              0
_________________________________________________________________
dense (Dense)                (None, 128)               802944
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
```

Figure 2: Output shapes and Number of parameters in each layer

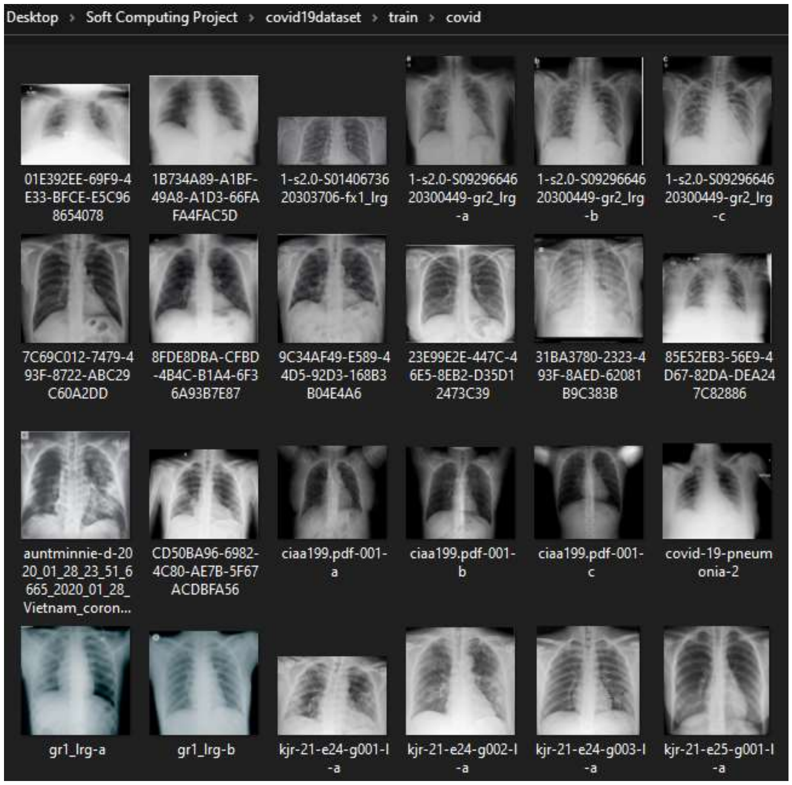

Figure 3: Sample images used in training.

This step is necessary because the x-ray samples used are of varying sizes and shapes. Sequential function is used to create our model layer by layer. In the progression of our model 7 layers are used in training of the model with 5 epochs. For metric calculation, accuracy and loss parameters are calculated for validation as well as training. At the end, trained model is saved into an h5 model so that results can be obtained from it without training it over and over.

Figure 4: Training of model in epoch 3/5

Layer Flow Chart:

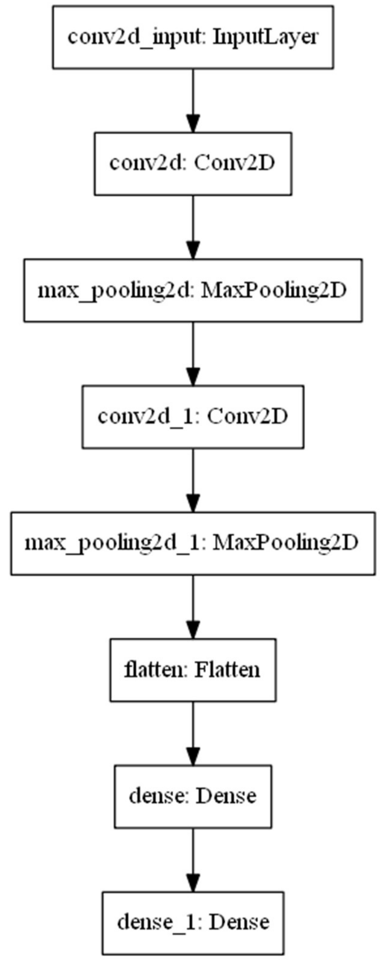

Figure 5: Flowchart of several layers making up CNN

Number	Layer	Activation Function
1	Conv2D	ReLu
2	MaxPooling2D	-
3	Conv2D	ReLu
4	MaxPooling2D	-
5	Flatten	-
6	Dense	ReLu
7	Dense	Sigmoid

Table 1: Activation function used in various layers.

Definitions:

Layers:

Conv2D:

Keras Conv2D is a 2D Convolution Layer, this layer creates a convolution kernel that is wind with layers input which helps produce a tensor of outputs. Kernel: In image processing kernel is a convolution matrix or masks which can be used for blurring, sharpening, embossing, edge detection, and more by doing a convolution between a kernel and an image.

MaxPooling

Max pooling operation for 2D spatial data. Downsamples the input representation by taking the maximum value over the window defined by pool_size for each dimension along the features axis. The window is shifted by strides in each dimension. The resulting output when using "valid" padding option has a shape(number of rows or columns) of: output_shape = (input_shape - pool_size + 1) / strides)

Flatten

Flattens the input image array

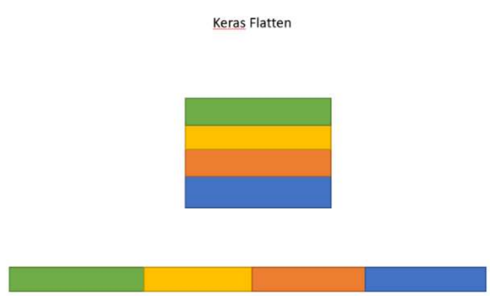

Figure 6: Schematic showing Flattening of and image array in Keras

Dense

Dense implements the operation: output = activation(dot(input, kernel) + bias) where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer, and bias is a bias vector created by the layer (only applicable if use_bias is True).

Activation Functions:

ReLU

The Rectified Linear Unit is the most commonly used activation function in deep learning models. The function returns 0 if it receives any negative input, but for any positive value x it returns that value back. So it can be written as

$$f(x)=\max(0,x)$$

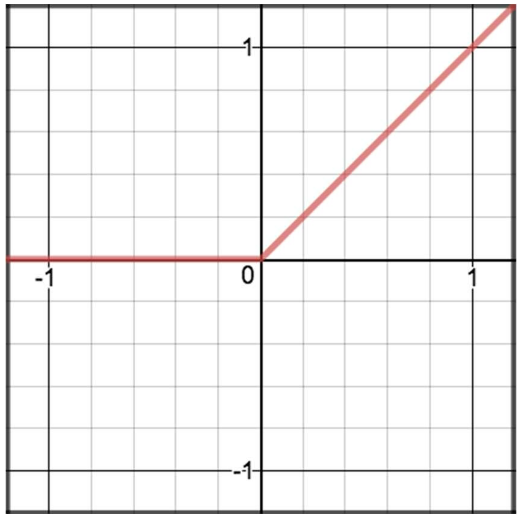

Figure 7: ReLU Function plotting

Sigmoid

A sigmoid function is a mathematical function having a characteristic "S"-shaped curve or sigmoid curve.

$$S(z) = \frac{1}{1 + e^{-z}}$$

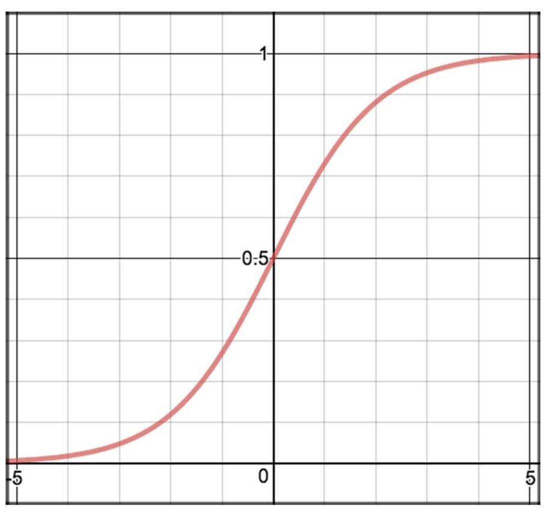

Figure 8: Sigmoid function plotting

Code:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import cv2
print()
train_image_generator = ImageDataGenerator(rescale=1.0/255)
test_image_generator = ImageDataGenerator(rescale=1.0/255)
training_images = train_image_generator.flow_from_directory(
                    'C:/Users/Param Prashar/Desktop/Soft Computing Project/covid19dataset/train',
                    target_size=(64, 64),
                    batch_size=8,
                    class_mode='binary')
testing_images = test_image_generator.flow_from_directory(
                    'C:/Users/Param Prashar/Desktop/Soft Computing Project/covid19dataset/test',
                    target_size=(64, 64),
                    batch_size=8,
                    class_mode='binary')
def plotImages(images):
    fig, axes = plt.subplots(1, 5, figsize=(20, 20))
    axes = axes.flatten()
    for img, ax in zip(images, axes):
        ax.imshow(img)
```

```
        ax.axis('off')
    plt.tight_layout()
    plt.show()
sample_training_images, _ = next(training_images)
# plotImages(sample_training_images[:5])
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
history = model.fit_generator(training_images, epochs=5, validation_data=testing_images)
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs_range = range(5)
plt.figure(figsize=(8, 8))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('ACCURACY')
```

```
plt.subplot(1, 2, 2)

plt.plot(epochs_range, loss, label='Training Loss')

plt.plot(epochs_range, val_loss, label='Validation Loss')

plt.legend(loc='upper left')

plt.title('LOSS')

plt.show()

model.save("model.h5")

model.save("model")

print(">> MODEL SAVED ")

-----------------------------------------------------------------------------------------------------------

from tensorflow.keras.models import load_model

import cv2

import numpy as np

model = load_model("C:/Users/Param Prashar/Desktop/Soft Computing Project/model.h5")

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# image = cv2.imread("C:/Users/Param Prashar/Desktop/Soft Computing
Project/covid19dataset/test/normal/NORMAL2-IM-1385-0001.jpeg")    # 1

image = cv2.imread("C:/Users/Param Prashar/Desktop/Soft Computing
Project/covid19dataset/test/covid/nejmoa2001191_f3-PA.jpeg")    # 0

image = cv2.resize(image, (64, 64))

image = np.reshape(image, [1, 64, 64, 3])

classes = model.predict_classes(image)

label = ["COVID-19 INFECTED", "NORMAL"]

print(classes)

print(label[classes[0][0]])
```

Outputs and Results:

```
[Command: python -u 'C:\Users\Param Prashar\Desktop\Soft Computing Project\B.py']

pciBusID: 0000:01:00.0 name: GeForce 940MX computeCapability: 5.0
coreClock: 1.2415GHz coreCount: 3 deviceMemorySize: 2.00GiB deviceMemoryBandwidth: 14.92GiB/s

Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class
[[1]]
NORMAL
[Finished in 10.834s]
```

Figure 9: Result showing NORMAL result when an image of unaffected lungs is fed.

```
[Command: python -u 'C:\Users\Param Prashar\Desktop\Soft Computing Project\B.py']

pciBusID: 0000:01:00.0 name: GeForce 940MX computeCapability: 5.0
coreClock: 1.2415GHz coreCount: 3 deviceMemorySize: 2.00GiB deviceMemoryBandwidth: 14.92GiB/s

Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class
[[0]]
COVID-19 INFECTED
[Finished in 8.017s]
```

Figure 10: Result showing INFECTED result when an image of infected lungs is fed.

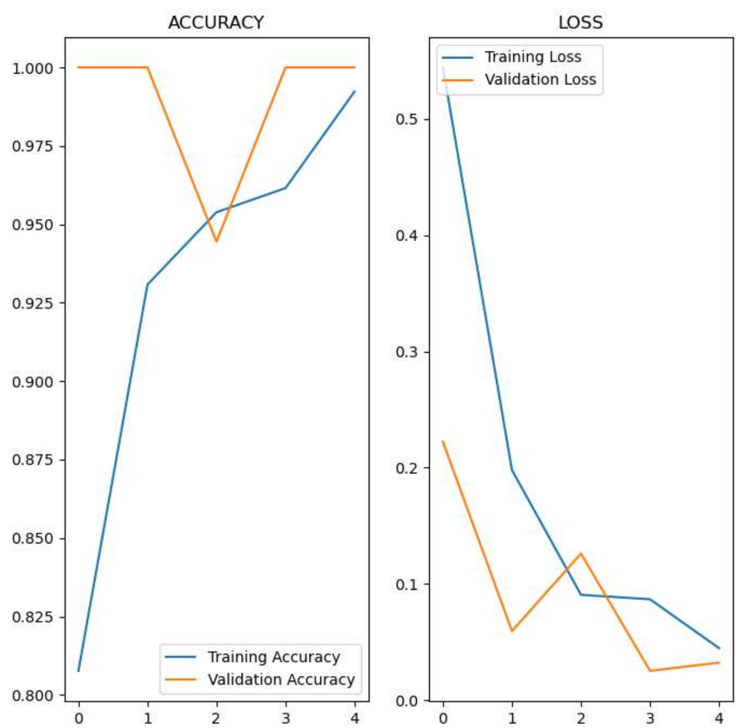

Figure 11: Accuracy and Loss plots for the model.

Final Metric Result:

Figure12: Actual screen shot showing metric results.

Classification Metric	Metric value
Loss	0.0447
Accuracy	0.9923
Validation Loss	0.0322
Validation Accuracy	1.0000

Table 2: Model Metric results.

Loss is defined as the difference between the predicted value by model and the true value. The most common loss function used in deep neural networks is cross-entropy. It's defined as:

$$\text{Cross-entropy } = -\sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j}\log\left(p_{i,j}\right)$$

$y_{i,j}$ denotes the true value i.e. 1 if sample i belongs to class j and 0 otherwise.

$p_{i,j}$ denotes the probability predicted by model of sample i belonging to class j.

Accuracy is one of the metrics to measure the performance of a model, defined as

Accuracy = (Number of correct predictions) / (Total number of predictions)

References

1. Sekeroglu, B. and I. Ozsahin. “Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks.” *Slas Technology* (2020): n. pag.
2. Islam, M. Z., M. M. Islam and A. Asraf. “A Combined Deep CNN-LSTM Network for the Detection of Novel Coronavirus (COVID-19) Using X-ray Images.” *medRxiv* (2020): n. pag.
3. Shibly, K. H., S. K. Dey, M. T. Islam and M. M. Rahman. “COVID Faster R-CNN: A Novel Framework to Diagnose Novel Coronavirus Disease (COVID-19) in X-Ray Images.” *medRxiv* (2020): n. pag.
4. Varshni, Dimpy, K. Thakral, Lucky Agarwal, Rahul Nijhawan and A. Mittal. “Pneumonia Detection Using CNN based Feature Extraction.” *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)* (2019): 1-7.
5. Yee, Sara Lee Kit and W. Raymond. “Pneumonia Diagnosis Using Chest X-ray Images and Machine Learning.” Proceedings of the 2020 10th International Conference on Biomedical Engineering and Technology (2020): n. pag.
6. https://kharshit.github.io/blog/2018/12/07/loss-vs-accuracy
7. https://towardsdatascience.com/building-a-convolutional-neural-network-cnn-in-keras-329fbbadc5f5