Monitoring Sustainable Development Goals Amidst COVID-19 Through Big Data, Deep Learning and Interdisciplinarity

MWITONDI, Kassim

Monitoring Sustainable Development Goals Amidst COVID-19 Through Big Data, Deep Learning and Interdisciplinarity

23rd-25th September 2020

Kassim S. Mwitondi (PhD) k.mwitondi@shu.ac.uk
URL: https://www.shu.ac.uk/about-us/our-people/staff-profiles/kassim-mwitondi
LinkedIn: https://uk.linkedin.com/in/kassim-mwitondi-9602091b
Twitter: @Mwitondi

Presentation Outline

1) Introduction
   i. Background, motivation, research question and objectives
   ii. United Nations Sustainable Development Goals (SDGs) and the global impact of COVID-19
   iii. Unified approach to addressing global challenges

2) Methods
   i. Structured Unstructured Data Sources
   ii. Interdisciplinarity and domain knowledge
   iii. Tools and techniques for visualisation and animation
   iv. Tools and techniques for predictive modelling

3) Analyses, results and discussions
   i. Demonstration: Tracking dynamics through animation in R
   ii. Demonstration: CNN architecture and training in Python
   iii. Practical issues for addressing global issues
   iv. Data randomness
   v. Uncovering interesting patterns in data. What do we need? Who knows what?
   vi. Construction of a CNN model and the Sample-Measure-Assess (SMA) algorithm

4) Concluding remarks
   i. Revisiting the objectives
   ii. Challenges, opportunities, success stories and suggestions for potential new research directions

Summary

As the coronavirus disease 2019 (COVID–19) ravaged across the globe, in 2020, the world was once again reminded of the gaps in our knowledge. The pandemic has had a severe impact on our ways of life, and despite its devastating impact, it has presented us with an opportunity for paying greater attention to the challenges we face. It is in that context that we associate the fight against COVID-19 with monitoring Sustainable Development Goals (SDG)-signed up by United Nations members, in 2015, to address global challenges–poverty, inequality, climate change, environmental degradation, peace and justice etc., by 2030 through measurable targets and indicators. Considering each SDG as a source of Big Data, we present a generic framework for combining Big Data, machine learning and interdisciplinarity to address global challenges. The work delivers descriptive and prescriptive findings, using data visualisation and animation techniques, on the one hand, and predictive results, based on convolutional neural networks, on the other. The former is based on structured data on cases and deaths from COVID–19 obtained from the European Centre for Disease Prevention and Control (ECDC) and data on the impact of the pandemic on various aspects of life, obtained from the UK Office of National Statistics. Predictive findings are based on unstructured data–a large COVID–19 X–Ray data, 3181 image files, obtained from Github and Kaggle. The results from both sets are presented in the form that resonates with cross disciplinary discussions, opening novel paths for interdisciplinary research in tackling global challenges.

Key words: *Big Data, Convolutional Neural Networks, Covid-19, Data Science, Data Visualisation, Interdisciplinarity, Predictive Modelling, Sustainable Development Goals*

Introduction

- The United Nations Sustainable Development Goals (SDGs) SDG [2] were signed up by 193 member states in 2015, outlining measurable targets and indicators to be attained by 2030. They seek to address the global challenges–poverty, inequality, climate change, environmental degradation, peace and justice, etc.

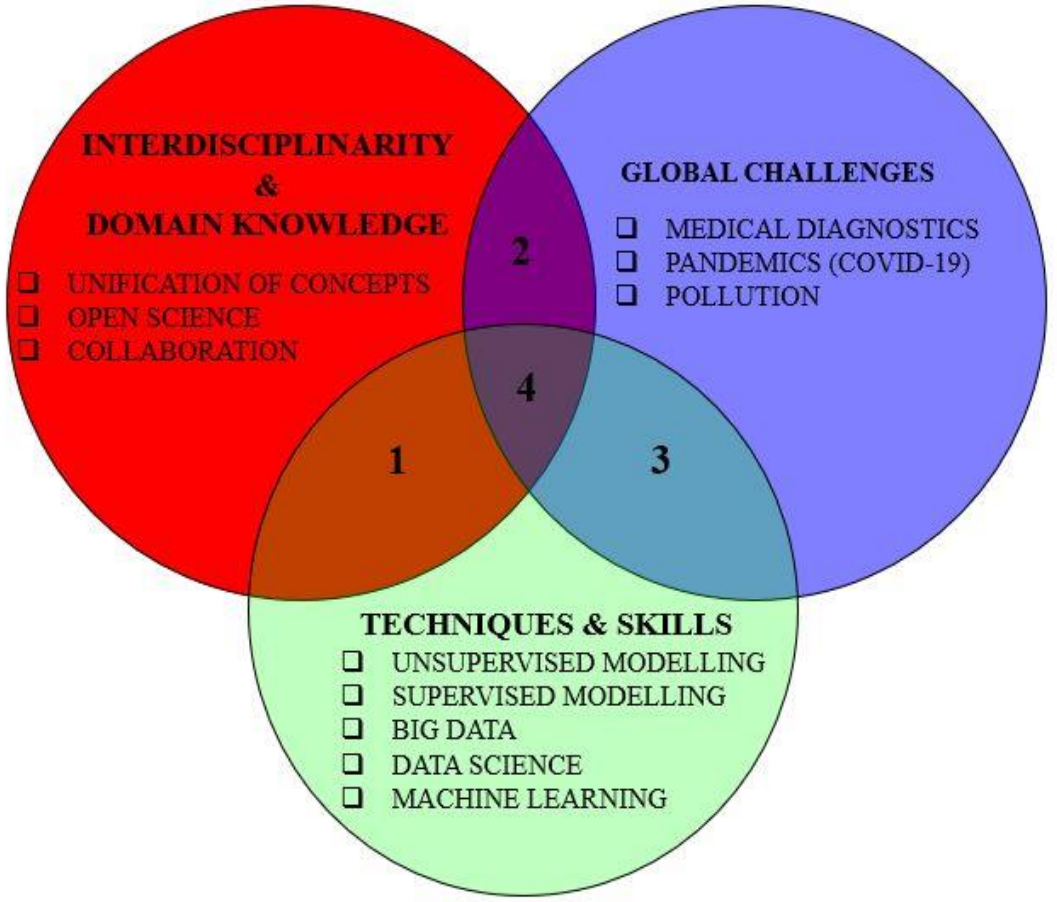
- The SDGs hugely over-lap, they span across sectors and regions. Their complex interactions, scope, magnitude and dynamics; their deep and wide socio–economic and cultural variations across the globe are both a challenge and an opportunity to the SDG project [8, 9]. Attainment requires a unified approach.

- Treating each SDG as a source of Big Data [8-12], this work seeks to highlight the path towards answering the question: **How can interdisciplinarity, Big Data and deep learning combine to deliver sustainable SDG solutions in the wake of the COVID–19 pandemic?** We set the following objectives.

1. To provide a descriptive mapping of skills and interdisciplinary knowledge for addressing global challenges.
2. To illustrate the impact of COVID–19 based on structured and unstructured data.
3. To demonstrate the efficacy of combining data, modelling techniques and skills in an interdisciplinary context.
4. To present analytical results through data visualisation, animation and predictive modelling.

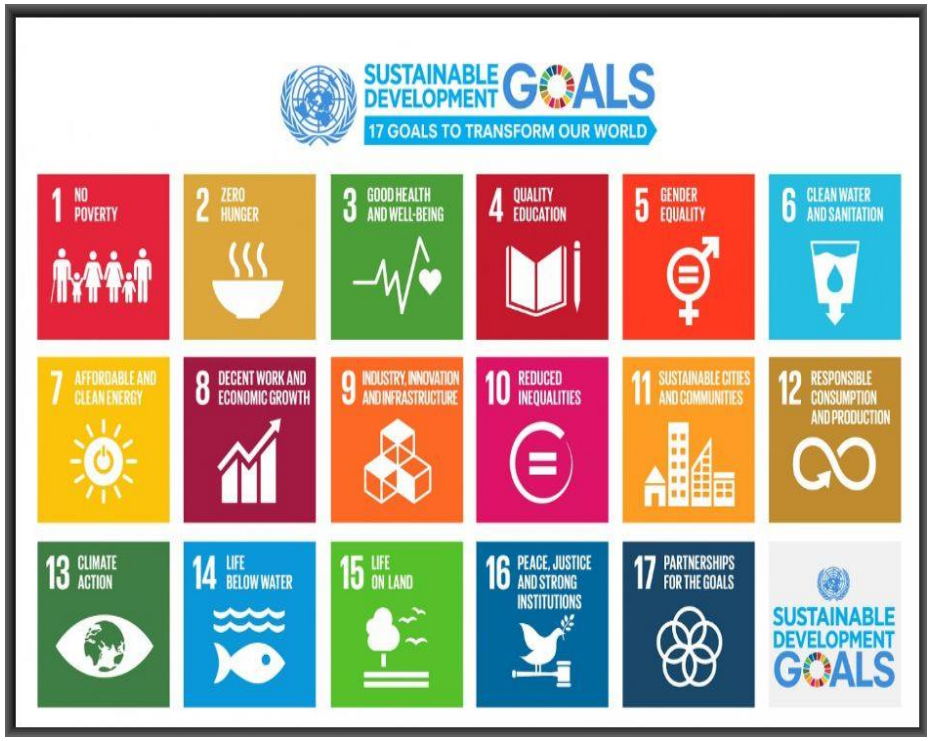
Key Interactions

- The intersections 1 through 4 are crucial.
- They resonate with the interdisciplinary approach to problem solving.
- Intersection #1 and #4 relate to aspects of data science
- Intersection #2 and #4 may relate to specific knowledge domains.
- Similar interpretations can be made for #1, #4 and #2 or the other combination of tripartites.
- It is from this perspective that the findings of this work are presented and discussed.



Big Data Modelling of SDGs (BDMSDG) - Structured Data

- While it is possible, to capture key metrics on indicators, their triggers remain buried in data.
- Any overarching strategy is complicated by the socio–economic, cultural and geo–political variations
- We need a unified understanding of the agenda at all levels.



WE TREAT EACH SDG AS A BIG DATA SOURCE

https://unstats.un.org/sdgs/indicators/database/

Big Data Modelling of SDGs (BDMSDG)

Advances in computing power and explosions in data generation, have triggered data–intensive research across disciplines, through, *inter–alia*, different applications aimed at addressing the challenges and opportunities of Big Data [1, 2, 3].

Across sectors & nations, Big Data brings challenges and opportunities from technical and application perspectives.

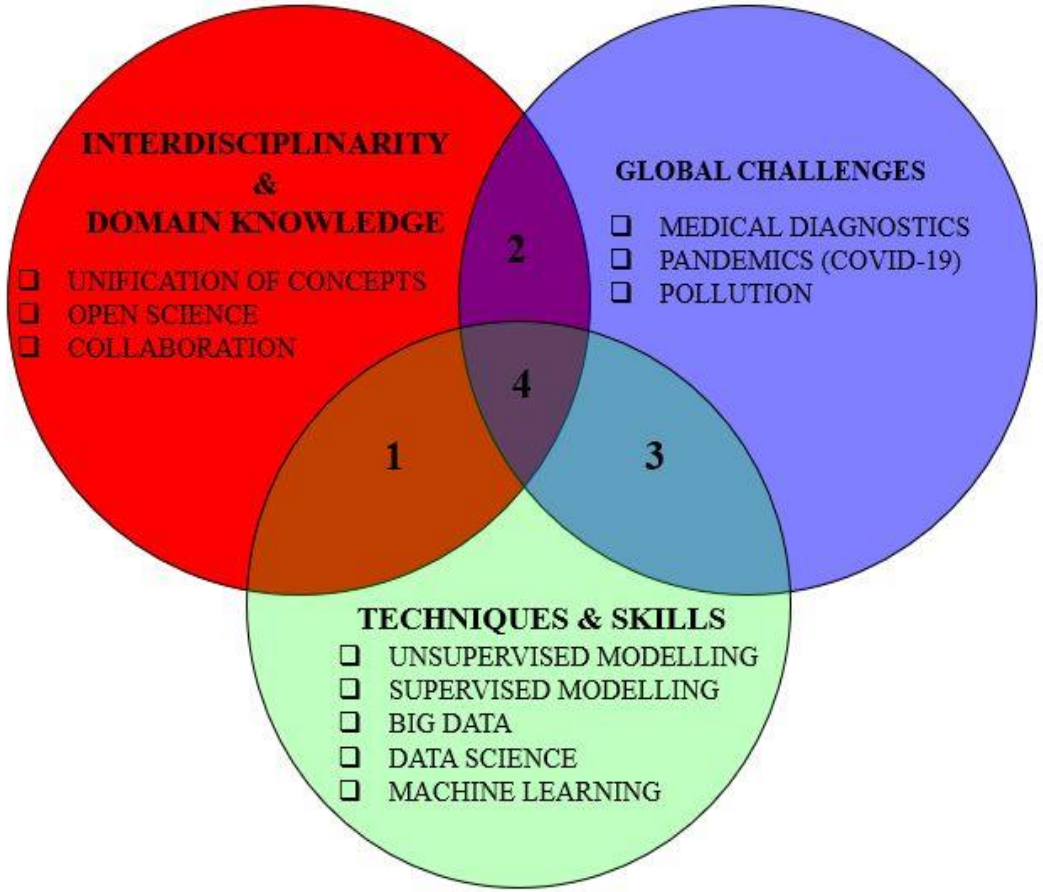
- Technically, they are pathways to addressing data sharing, modelling, infrastructure, security, governance.
- Application-wise, they relate to influential policies for improving decision making at institutional, national, regional and global levels, sustaining development.
- They present potential knowledge for unlocking our understanding of the mutual impact–positive and negative, resulting from our interaction with our environment. Resonating with climate change.

Our sustainability depends on our adaptive understanding of the triggers of known and potential positive and negative phenomena we face - species facing extinction, hunger and poverty, low productivity, land degradation, gender inequality or gaps in education quality and technological achievements span across sectors and regions.

Methods: Data Sources

The interactions in the Venn diagram imply that there will be **multi-faceted data** from various sources. To address the challenges we face, we need a unified understanding of the underlying concepts.

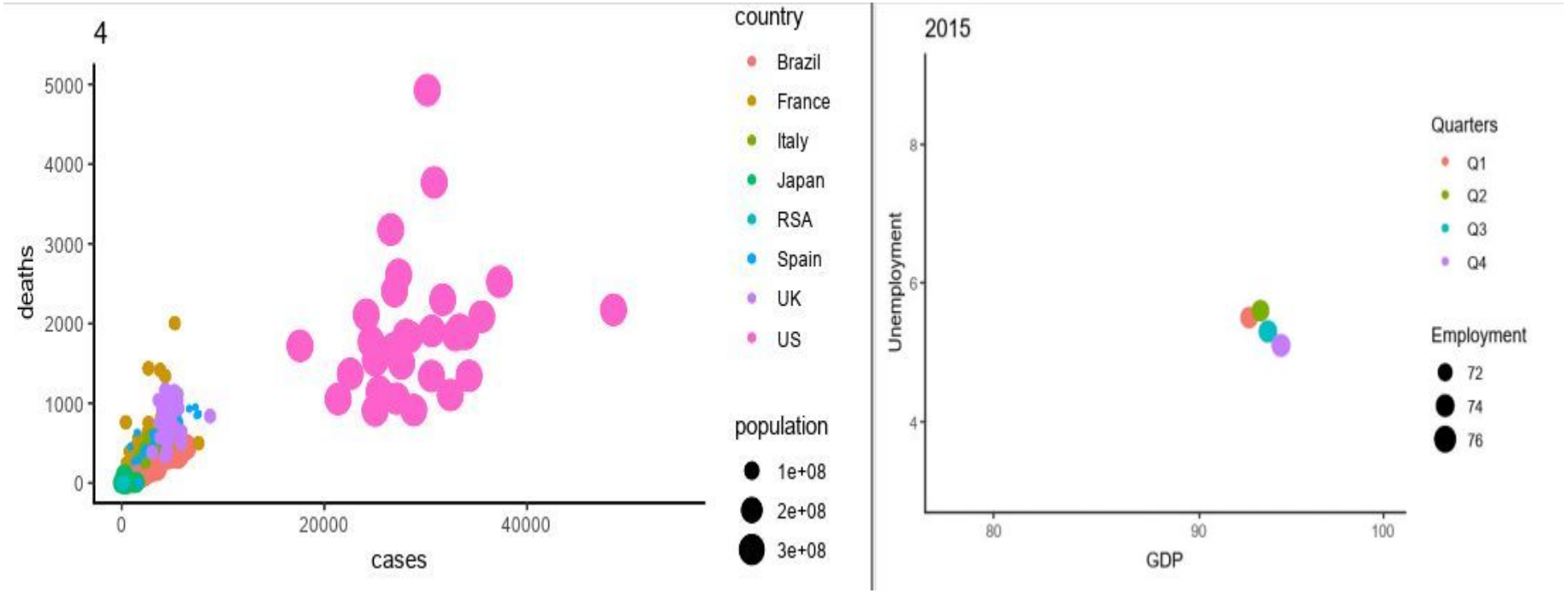
- Structured data came from the European Centre for Disease Prevention and Control (ECDC)[13] and the UK Office of National Statistics[14]. The former provides daily updates on cases and deaths per country based on a 14-day notification rate of new COVID-19 cases and deaths. The latter provided multiple data files on business, industry and trade as well as on the general economy and on the dynamics on the labour market before and during the pandemic.
- Unstructured training and validation datasets were obtained from Github[15] and Kaggle[16] - a large COVID-19 positive X-Ray data, 1840 images and 1341 non-COVID-19 data.



Methods: Data Sources

Impact of COVID-19

Global Cases and Deaths



UK Economic Impact

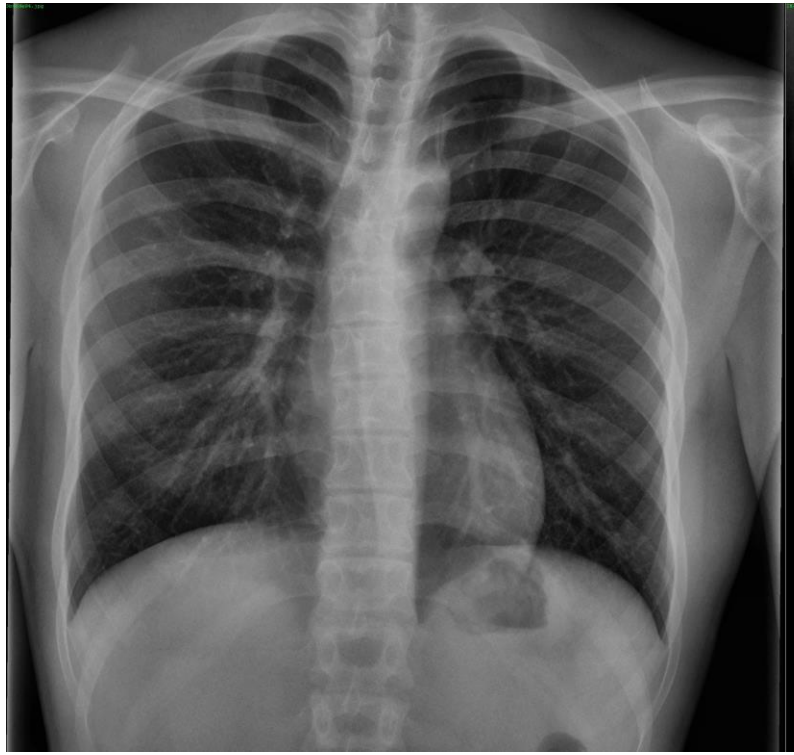
Animated in R and updatable in near real time...

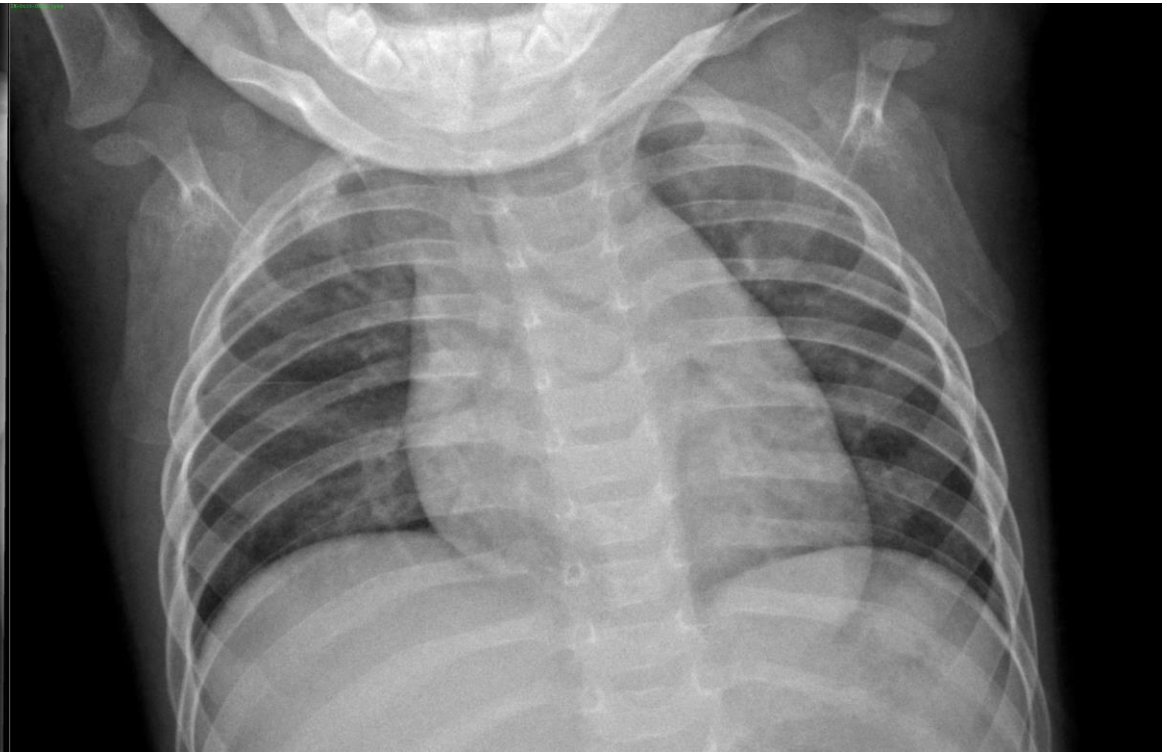
Methods: Data Sources

Imagery (X-Ray) data

Covid-19 Positive

https://github.com/ieee8023/covid-chestxray-dataset





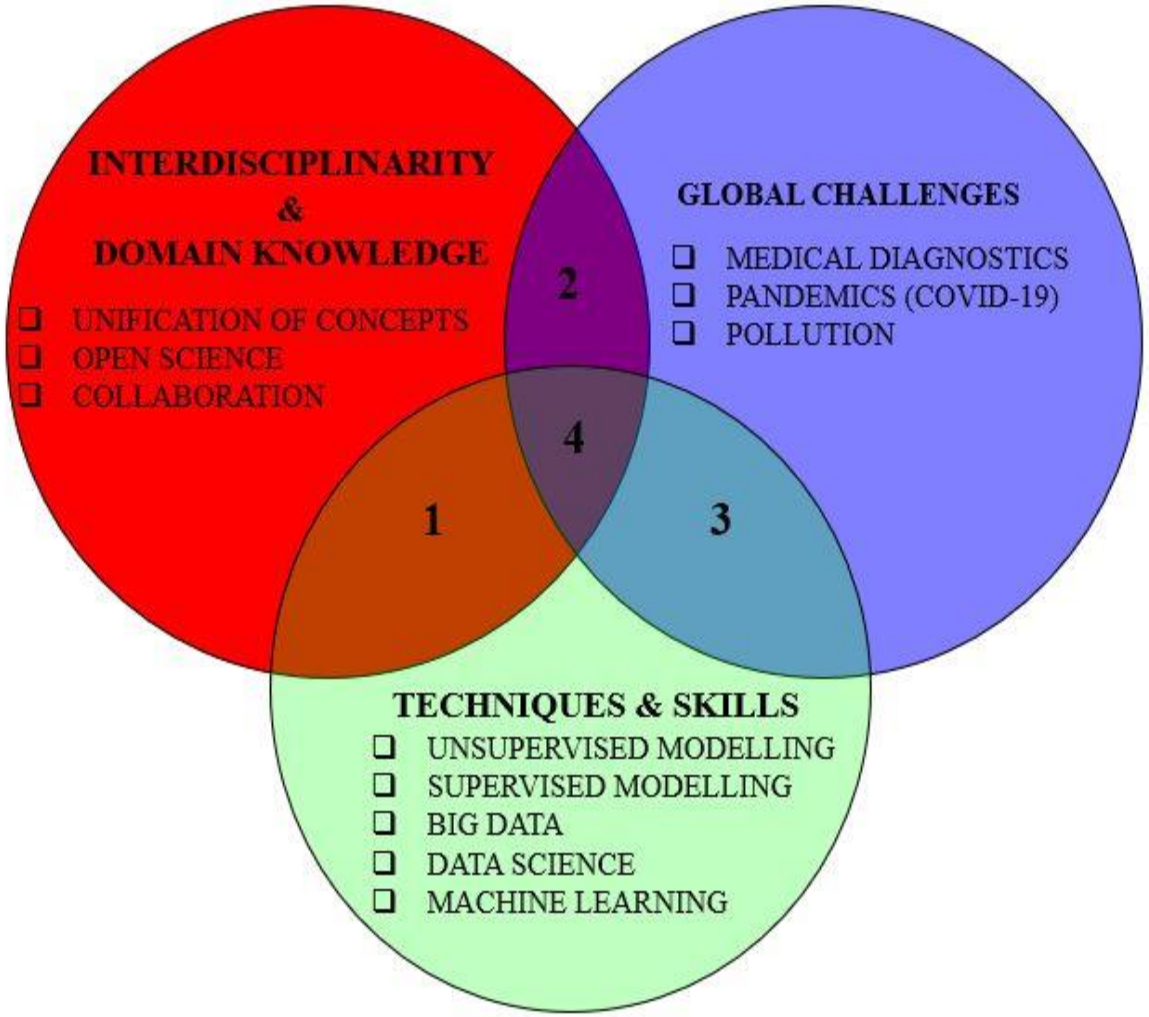
Non-Covid-19

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

Methods: Techniques

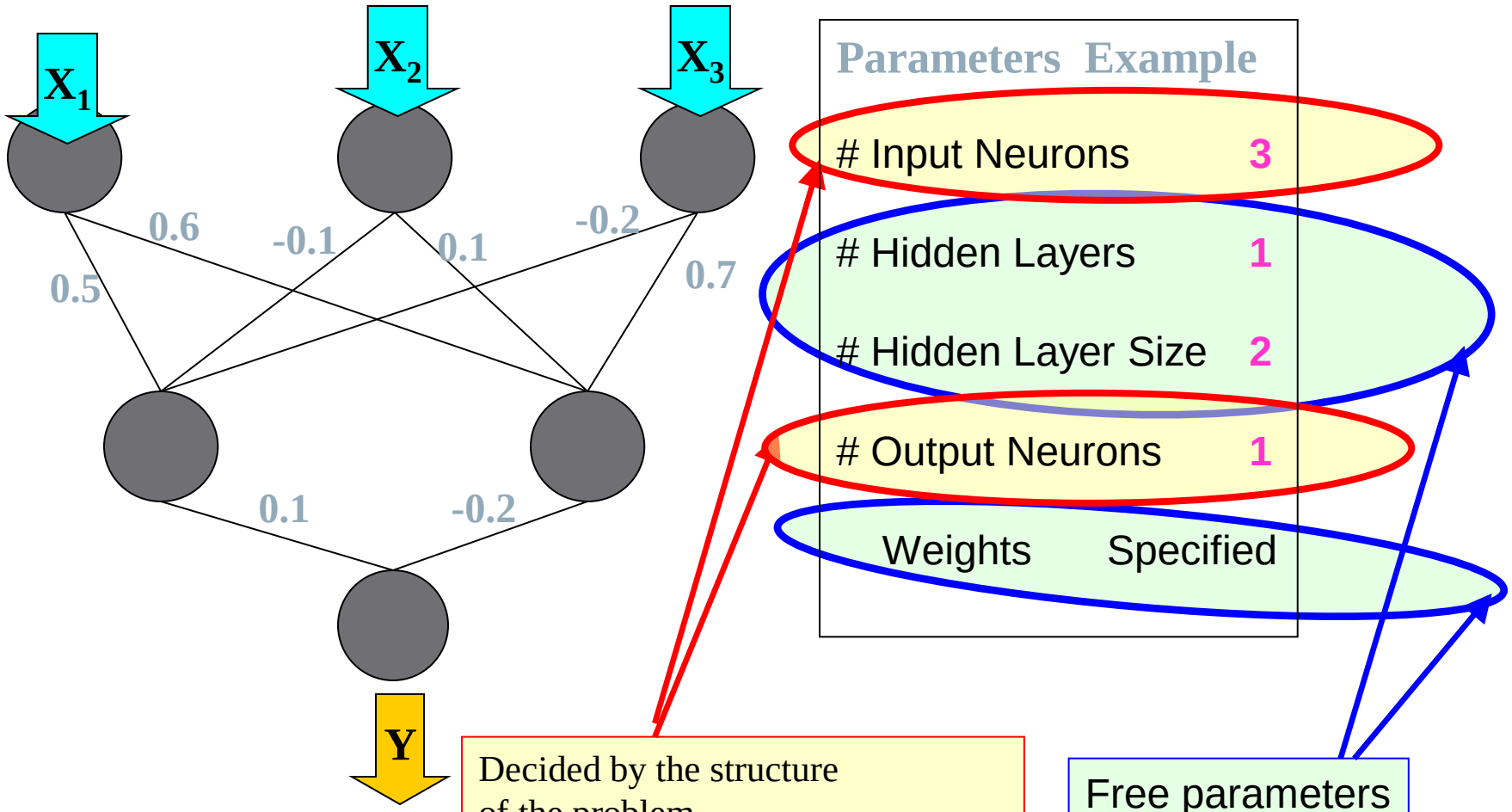
Extracting knowledge from data requires a combination of tools, techniques and interpretations guided by domain knowledge. In this work we adopt techniques in R and Python.

- **Visualisation and animation** of structured data provide an opportunity for tracking phenomena in time and space. They are handy in delivering digestible information to non-technical people.
- **Visualisation and animation** can help guide influential policies for improving decision making at institutional, national, regional and global levels, hence sustain development.
- **Convolutional Neural Networks (CNN):** We build and train a CNN model in Python to perform predictive analysis of unstructured data.
- **CNN** present an enhanced image analysis, playing the role of radiologists. Appropriately applied, CNN present great enhancement in our understanding of the biomedical challenges.



Methods: Techniques (Artificial Neural Networks)

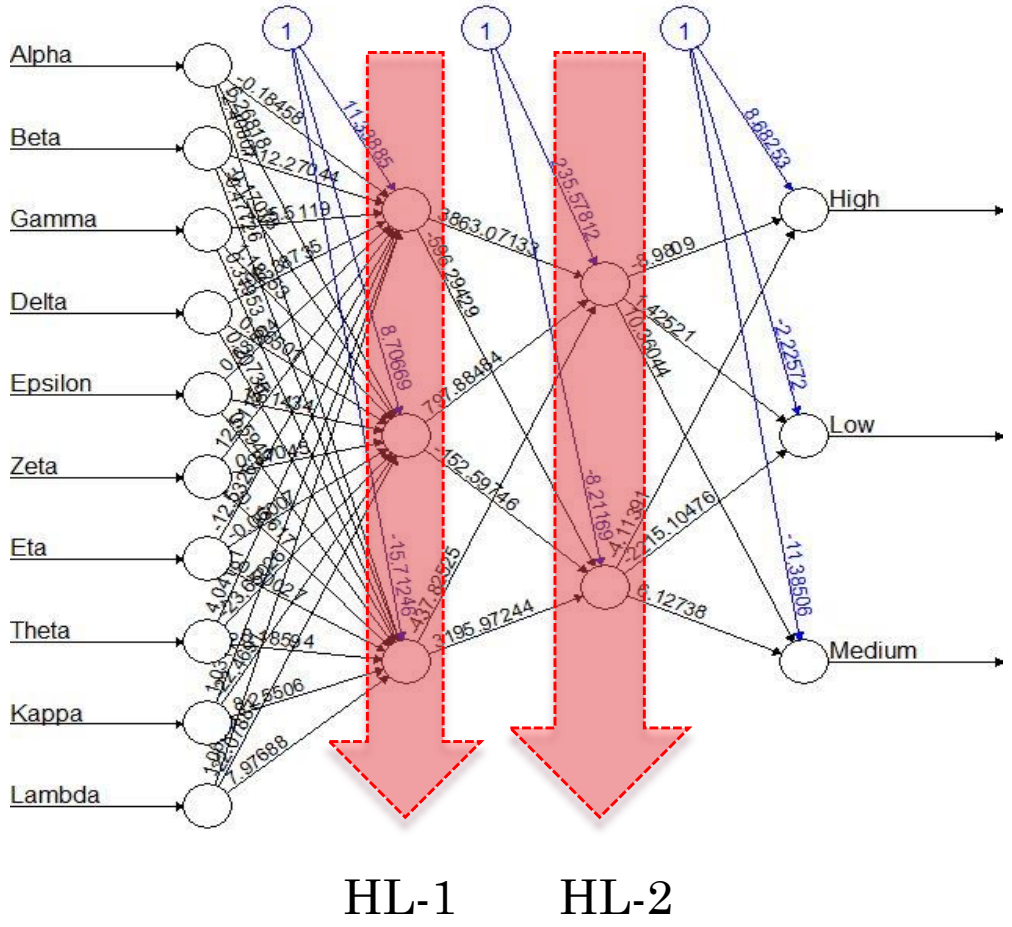
Input: X_1 X_2 X_3 **Output:** Y **Model:** $Y = f(X_1 X_2 X_3)$



Convolutional Neural Networks for Imagery Data

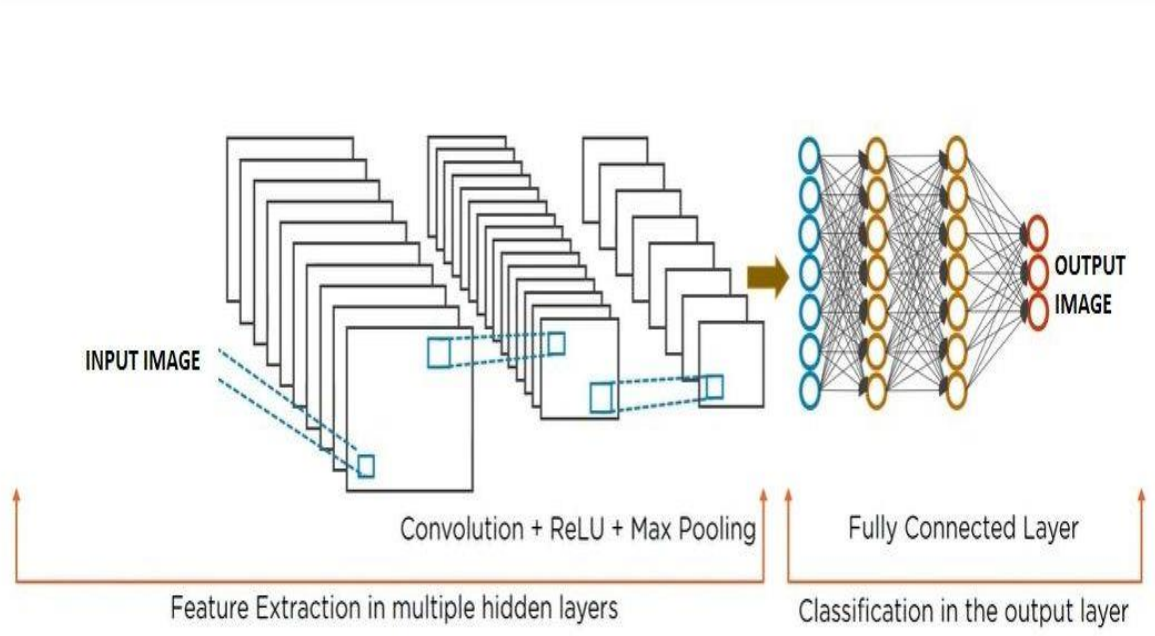
- Neurons at HLs are one dimensional while with CNN, they are 3 dimensional (Width, Height and Depth). The role of hidden layers in CNN is played by Convolutions. Unlike ANNs, CNNs don't require full connectivity.

- Curse of dimensionality: A 20x20 image would require 1200 sets of weights-that is 20x20x3 (RGB). Convolution 1 will transform... Pass its output on to convolution 2 which will process and generate the output, in this case, an image.



Convolutional Neural Network Architecture

- **Input Image:** This is, in our case the two COVID-19 scenarios – positive and negative
- **Feature Extraction:** Basically hidden layers – they consist of convolution, ReLU and Max Pooling.
- **Fully Connected** layer then receives this as input, for classifying the image…



- There are 3 types of these layers, namely: **Convolutional (CL), Pooling (PL)** and **Fully Connected (FC). Pooling** layers lie between CLs and aim to reduce parameters & improve robustness, by retaining only the most important features.
- The main idea of CNN is to "convolve" an image with a filter aimed at extracting the most important features from the image. That is to so what a radiographer would do with a naked eye.
- Think of CNNs as a sequence of layers each transforming a volume of activations to another through a differentiable functions (partial derivatives) and ultimately connecting the output and input.

Methods: Activation Functions – the Engines of ANN/CNN

- The sigmoid function returns 1 for all positive large inputs and 0.00 for all values less of equal to zero while the hyperbolic tangent function values are [-1.0, 1.0].

- Large networks using these functions tend to lose useful gradient information as the error is back-propagated into the network. Each additional layer tends to decrease the back-propagated error and as this is determined by the partial derivatives of the weights, it causes a problem commonly referred to as vanishing gradient.

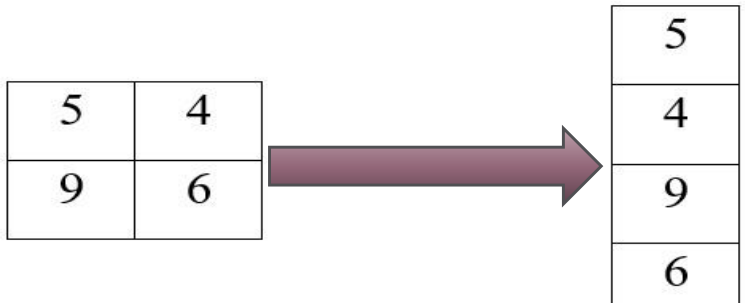
- The weight in a CNN play a similar role as coefficients in a linear regression model. It expresses the rate of change in the total loss would change, as the weight changes by one unit – i.e., the derivative of the Loss (L) with respect to W.

$$\frac{\partial L}{\partial w} = \lim_{\partial w \to 0} \left[\frac{L(w + \partial w) - L(w)}{\partial w} \right]$$

Rectified Linear Unit ReLU

- The **pooling** layer reduces the dimensionality of the rectified feature map. It uses different filters to identify different parts of the image – like edges, corners, curves etc.

- **Flattening** converts the 2-D arrays from the pooled layer into a one-dimensional vector.



- The **Fully Connected** layer then receives this as input, for classifying the image. The Rectified Linear Unit (ReLU) applies the activation function, the most common of which being

$$f(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{otherwise} \end{cases}$$

Maxi

$$f(x) = \frac{1}{(1 + e^{-x})}$$

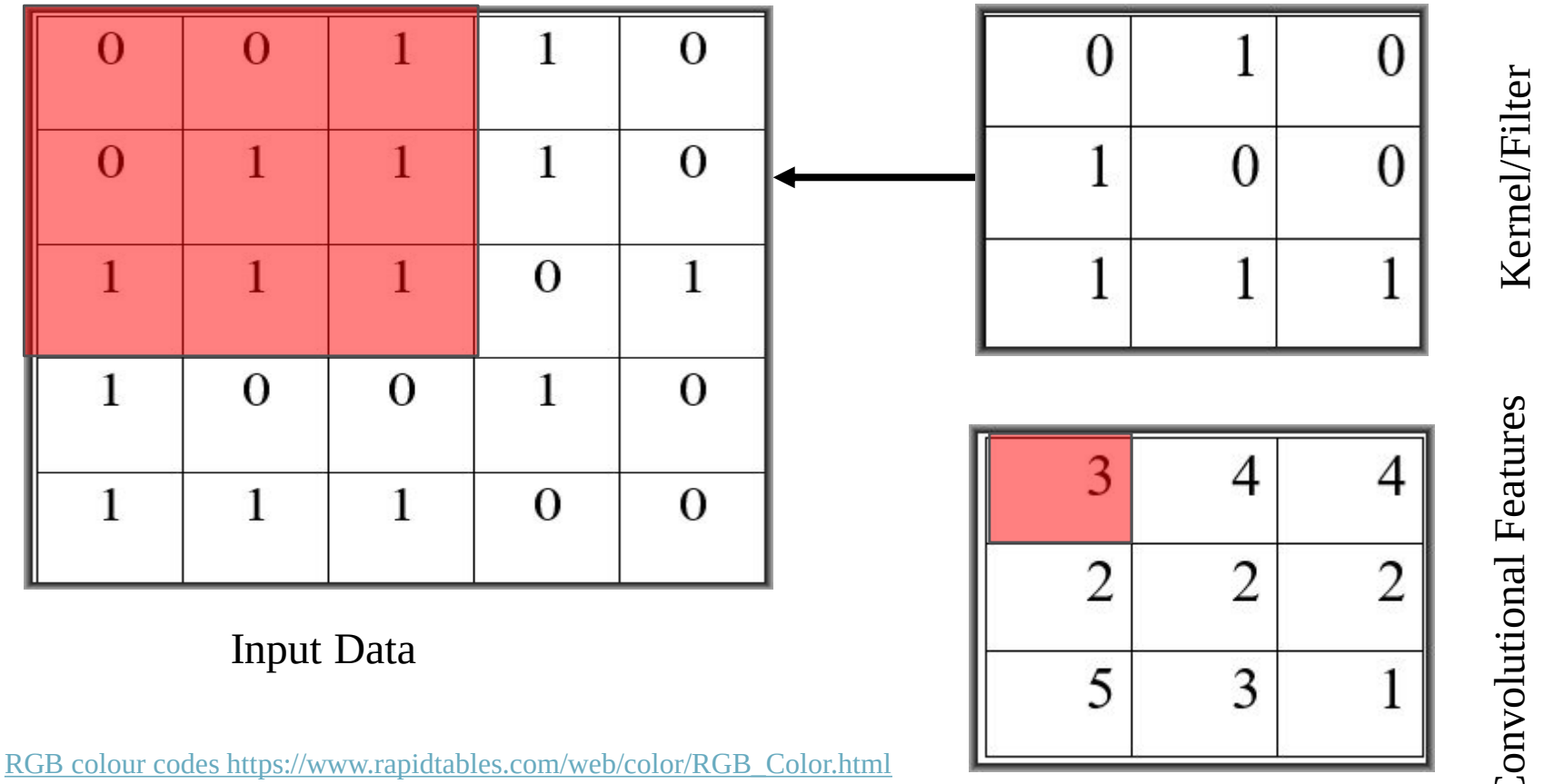
Sigmoid

- Maxi is a piecewise linear function that outputs the input directly if is positive or zero otherwise. In other words, it is a binary rule that outputs the maximum positive number or zero otherwise.

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.

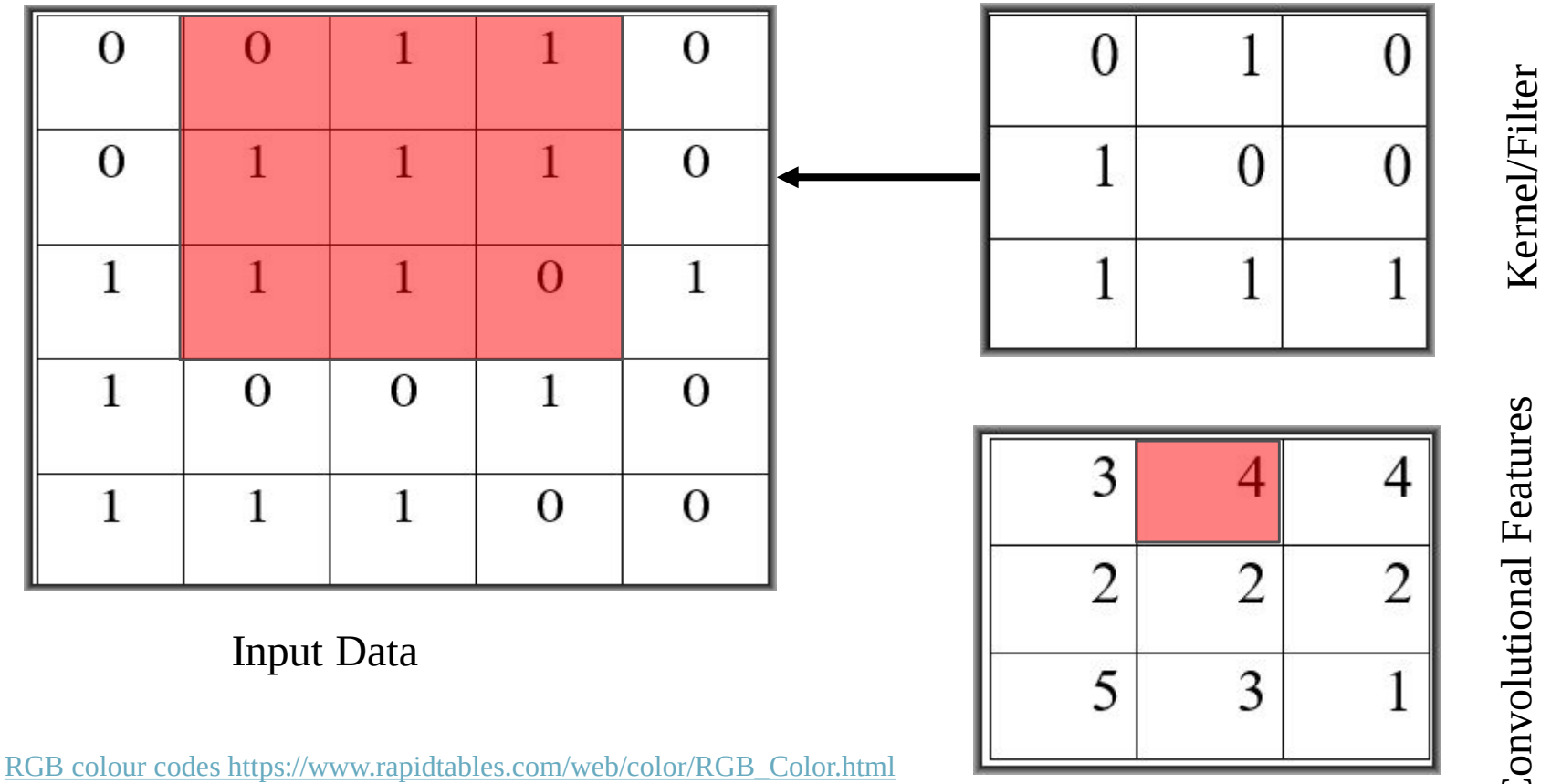


RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.



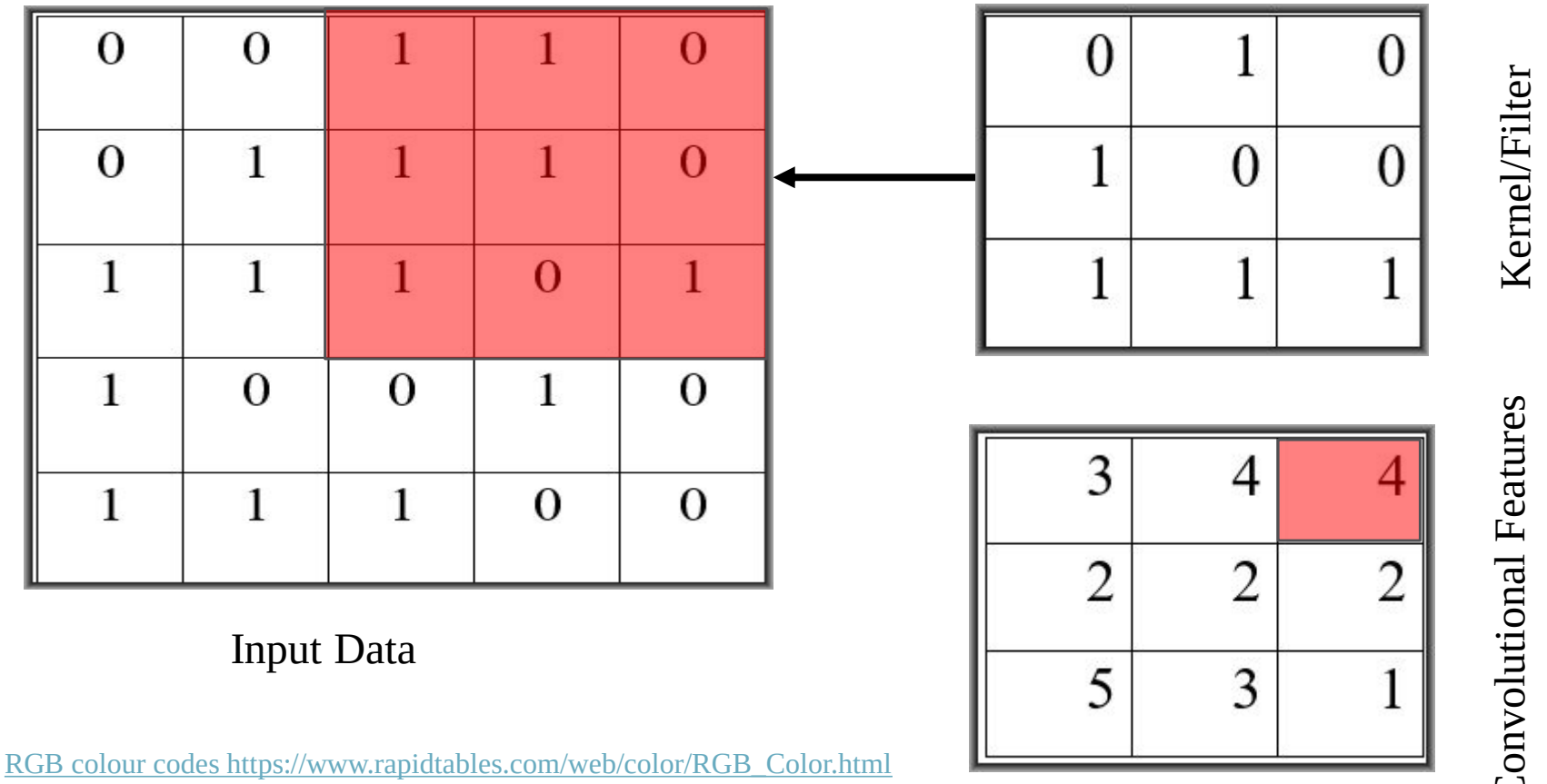
Input Data

RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.



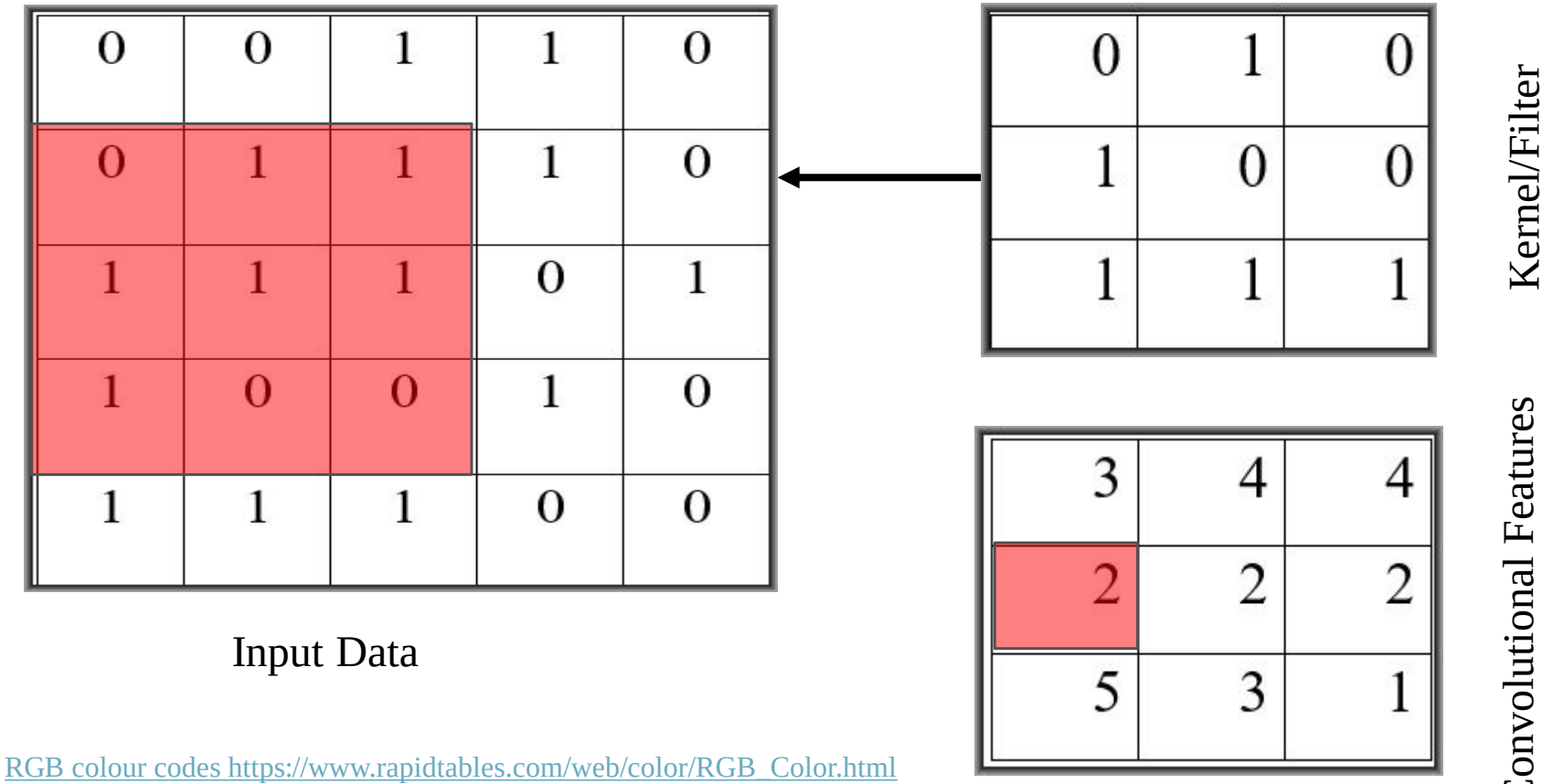
Input Data

RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.



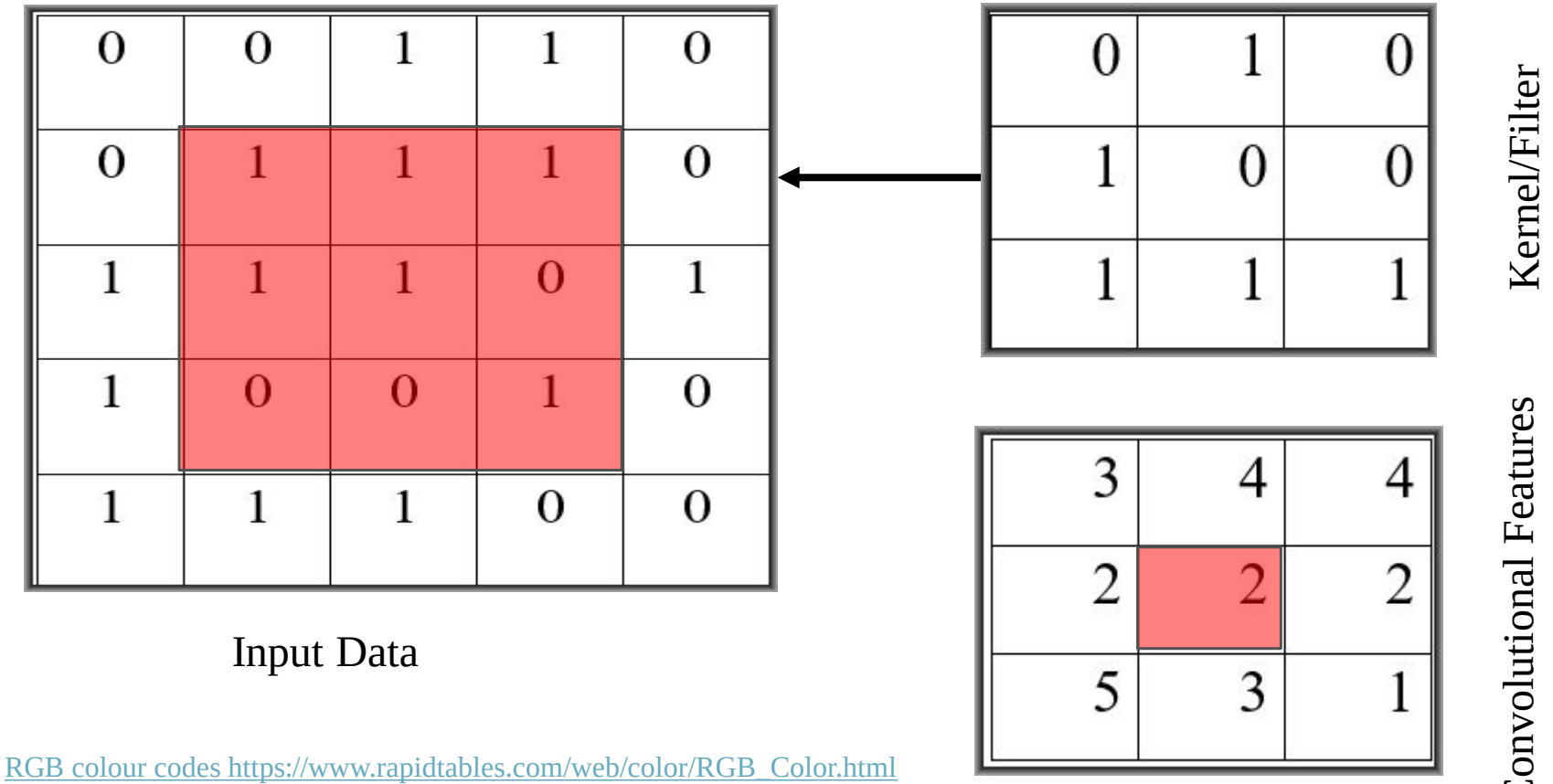
Input Data

RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.



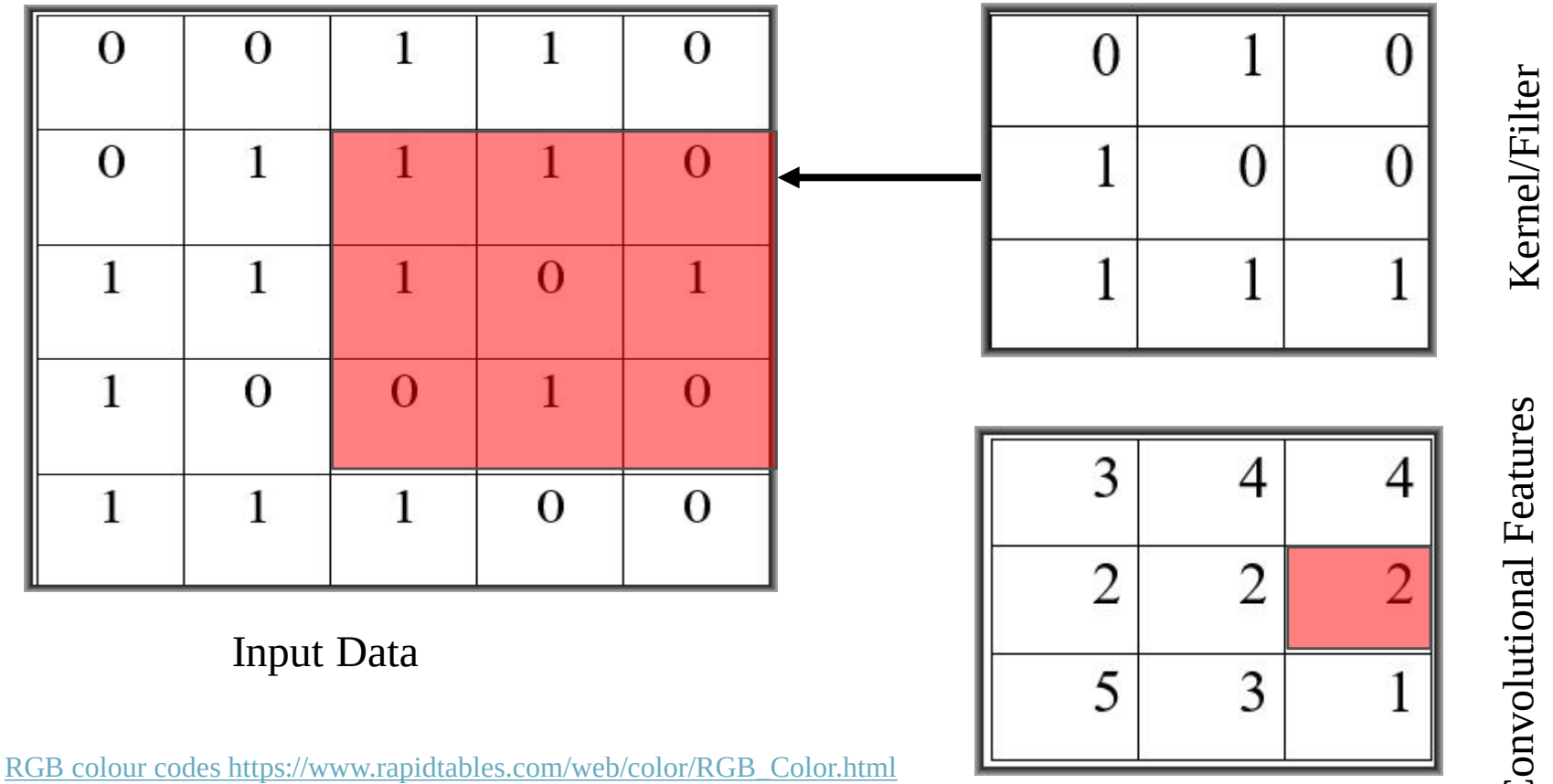
Input Data

RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.



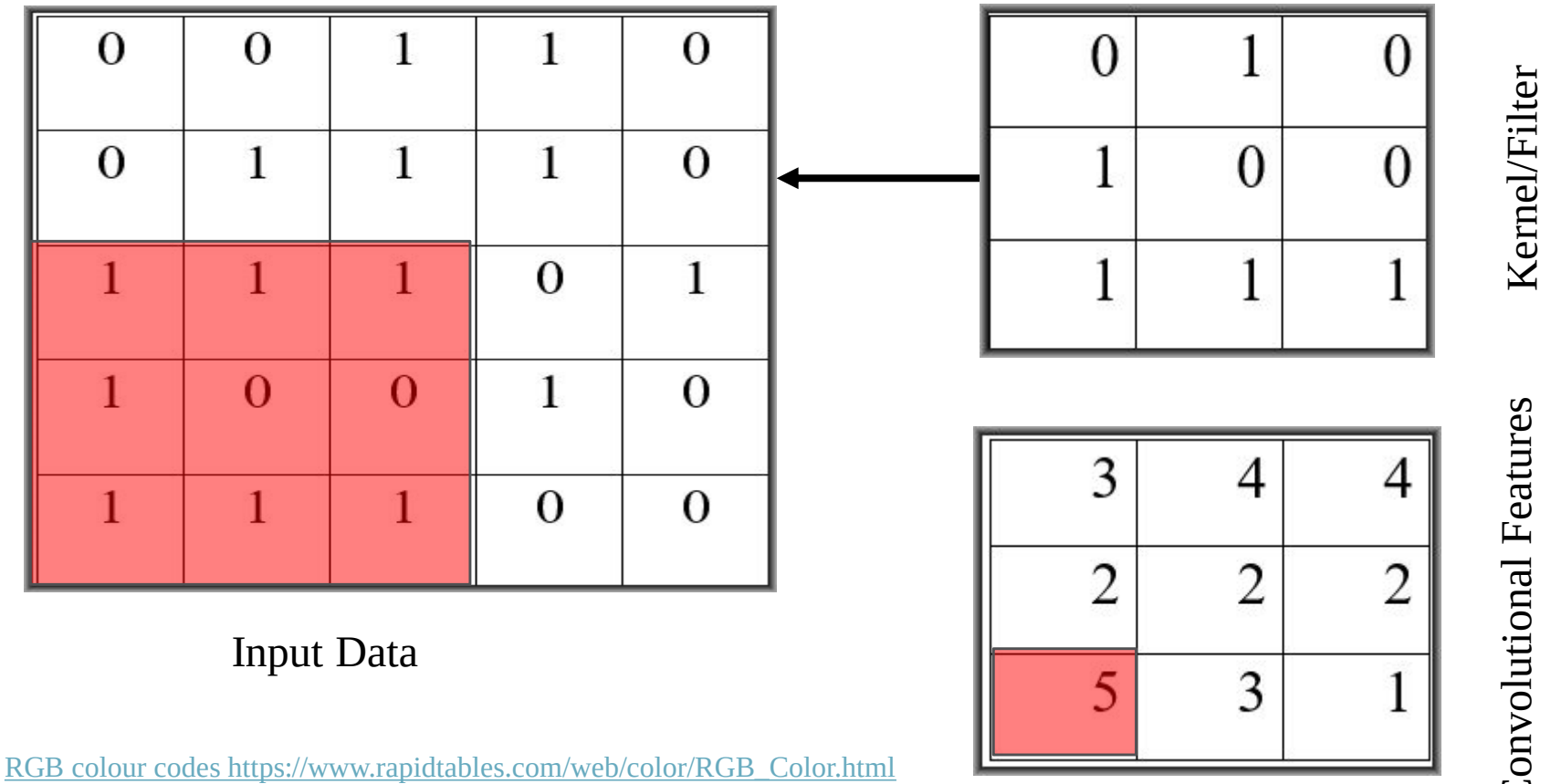
Input Data

RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.

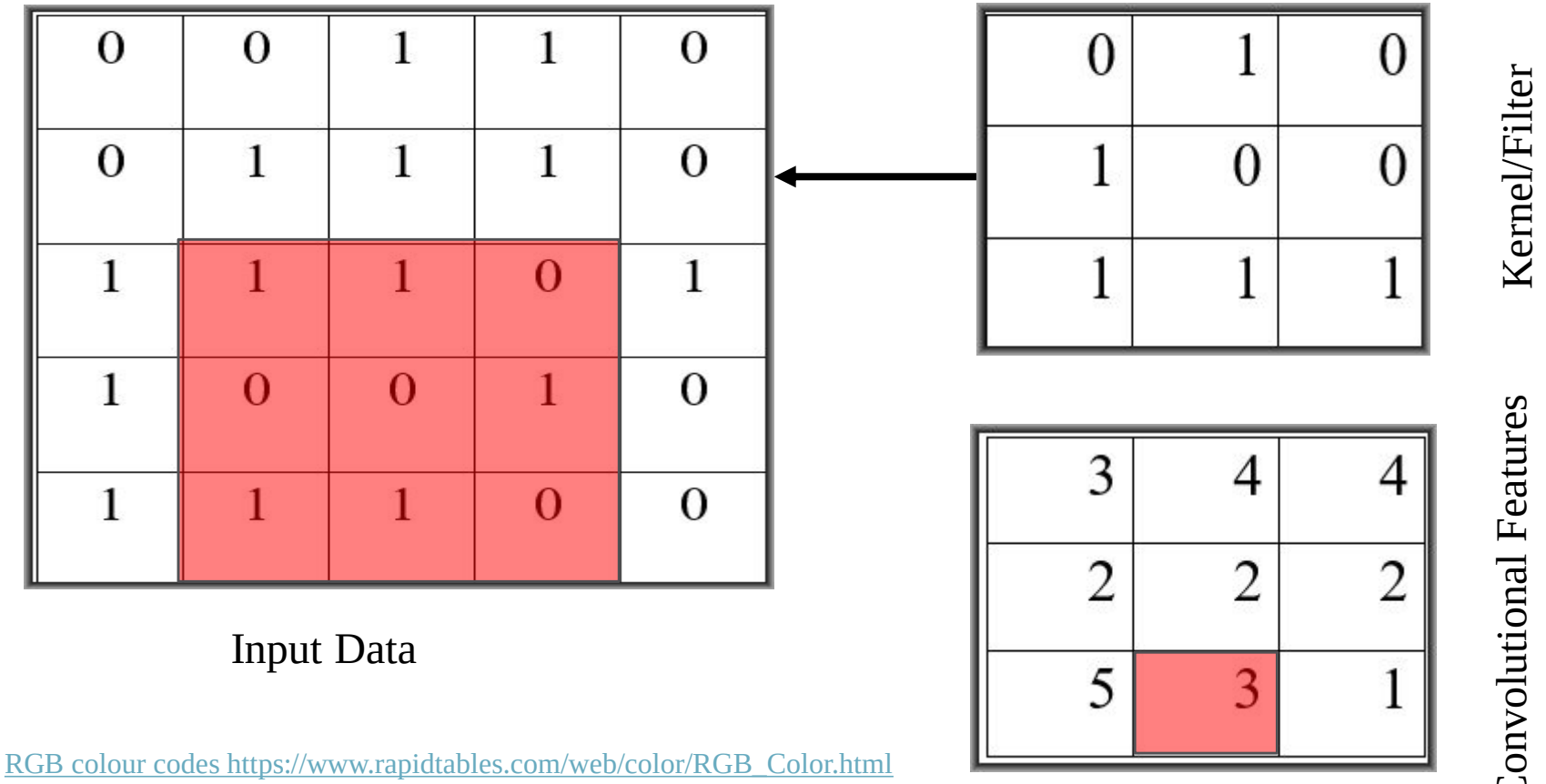


RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.



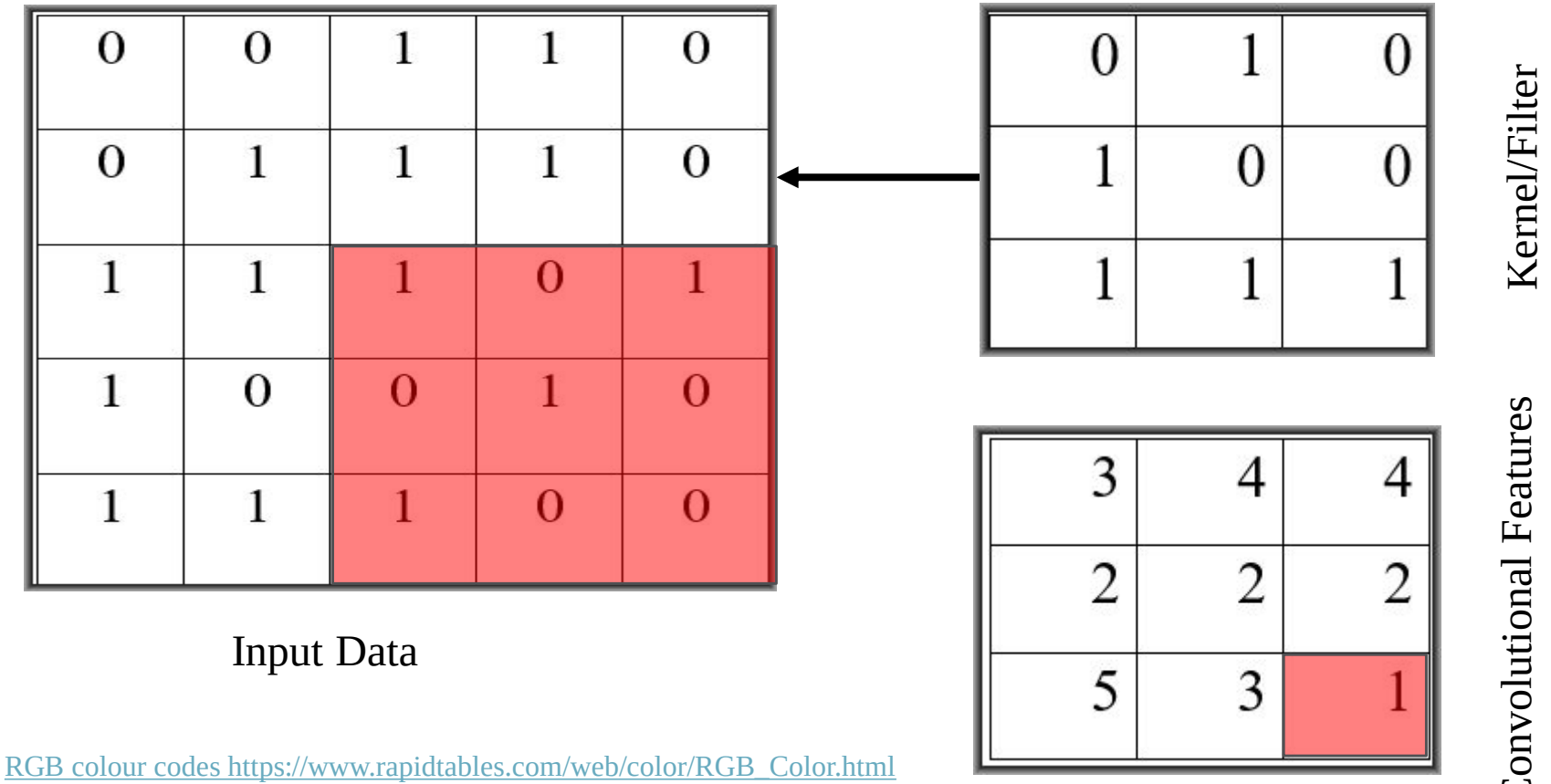
Input Data

RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

Using the kernel to extract convoluted features from input data

To get the convolutional values, slide the kernel over the input data, multiplying the corresponding values and summing up and fill the matrix of the same dimension as the kernel.

Note that the filter has reduced the input matrix to a smaller dimension of its own size.

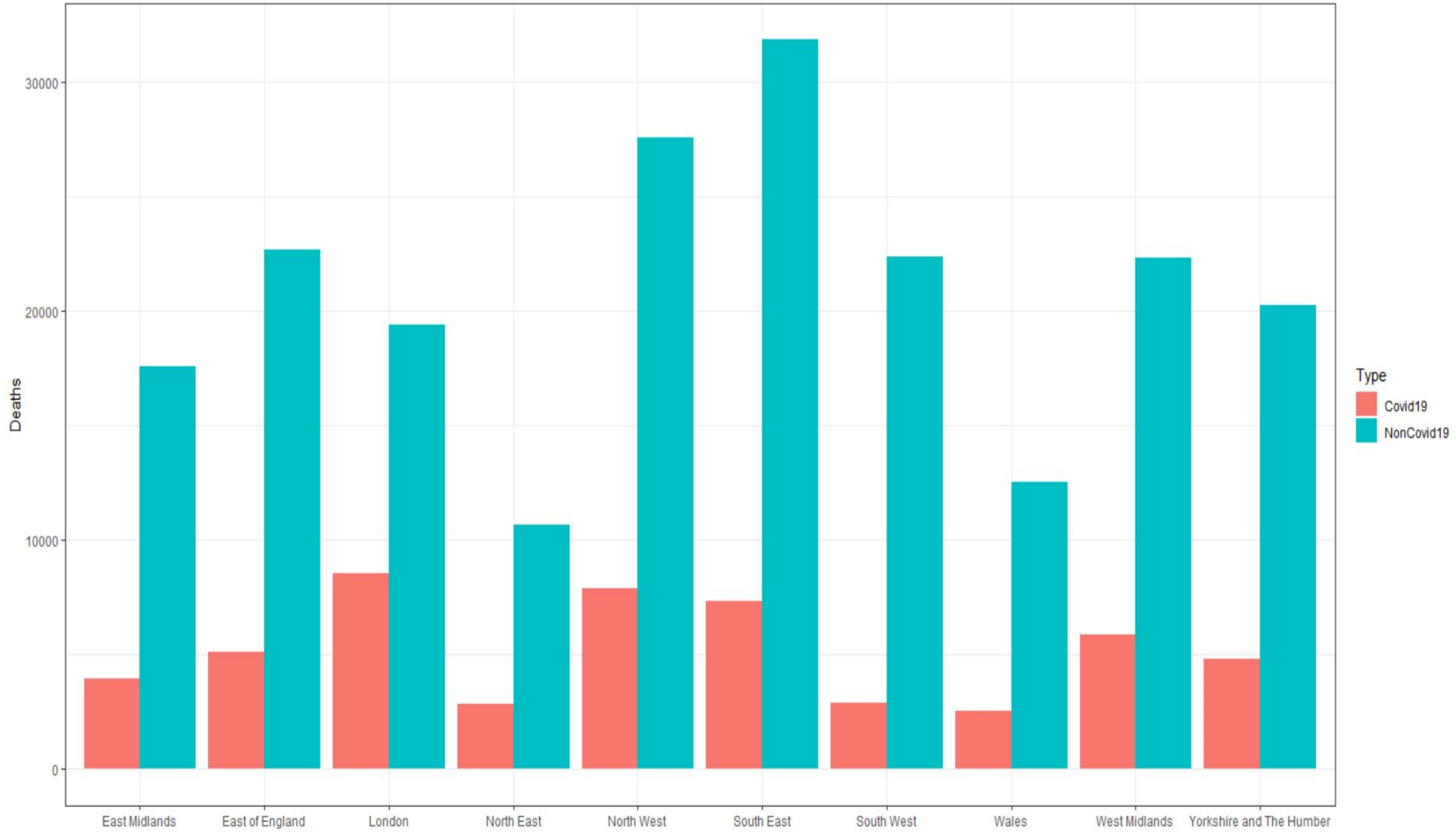


RGB colour codes https://www.rapidtables.com/web/color/RGB_Color.html

The Kernel

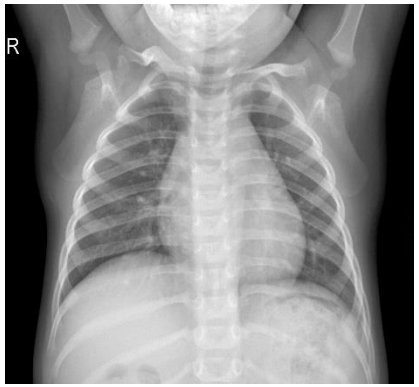
- A 2x2 pooling layer, filtering with a sliding of 2 down-samples at every depth of the input discards 75% of the activations.
- In this example, we applied a 2D convolution filter (***m*** *x* ***m***), which has a third dimension, equal to the number of channels of the input image. For the grey-scale X-Ray images, it is ***m*** *x* ***m*** x1 (black and white channel), whereas for colour images, it is ***m*** *x* ***m*** *x 3* (RGB).
- Changing the dimension of the kernel has an impact on the way features are learnt on the image under study.

Analyses: Structured Data



Analyses: Predictive Modelling of Unstructured Data

XRAY-INPUT



PROCESSING

OUTPUT: Normal

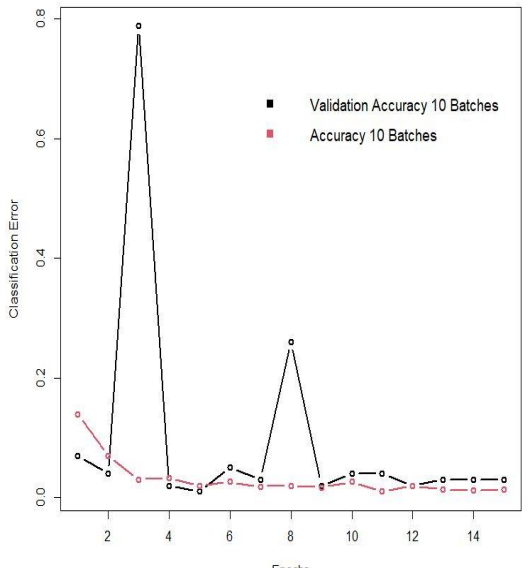
OUTPUT: Covid-19

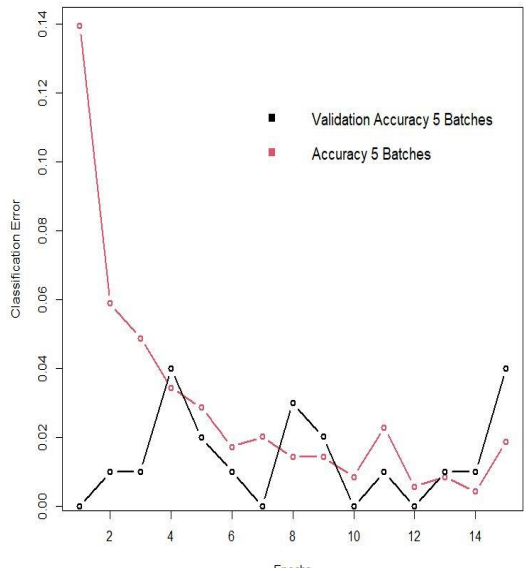
Analyses: CNN Model, Classification Error and Loss

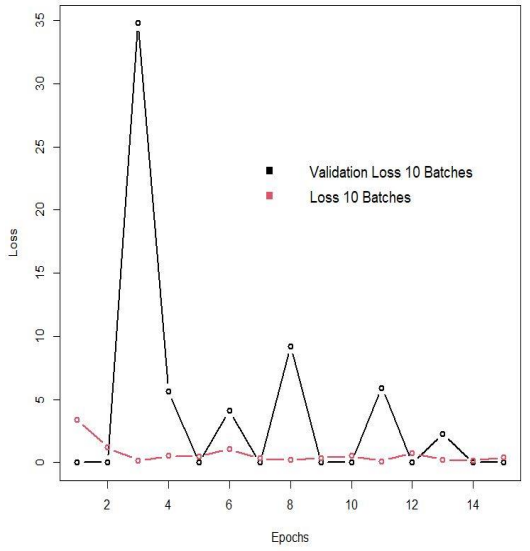
In [9]:
```
datagen = tf.keras.preprocessing.image.ImageDataGenerator(validation_split=0.3)
# load and iterate training dataset
image_width, image_height = 100, 100
trainsamples=700
validsamples=100
testsamples=150
epochs=15
batchsize=5

training = datagen.flow_from_directory("G:/ACADEMIC AND RESEARCH-RELATED/JAPAN-2020/covid-chestxray-dataset-master/modelling", cl
validation = datagen.flow_from_directory("G:/ACADEMIC AND RESEARCH-RELATED/JAPAN-2020/covid-chestxray-dataset-master/modelling",

if K.image_data_format()=='channels_first':
    inputshape=(3, image_width, image_height)
else:
    inputshape=(image_width, image_height, 3)
train_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_f
test_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
if K.image_data_format()=='channels_first':
    inputshape=(3, image_width, image_height)
else:
    inputshape=(image_width, image_height, 3)
train_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_f
test_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
```

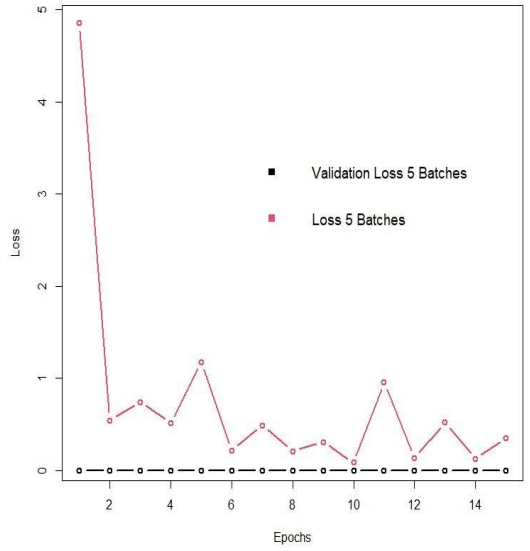
```
Found 651 images belonging to 2 classes.
Found 279 images belonging to 2 classes.
```

In [10]:
```
model=Sequential()
model.add(Convolution2D(32,(3,3),input_shape=inputshape))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Convolution2D(32,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Convolution2D(64,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(64))
model.add(Activation("relu"))
model.add(Dropout(0.25))
model.add(Dense(1))
model.add(Activation("sigmoid"))
model.summary()
```









Loss vs Validation Loss

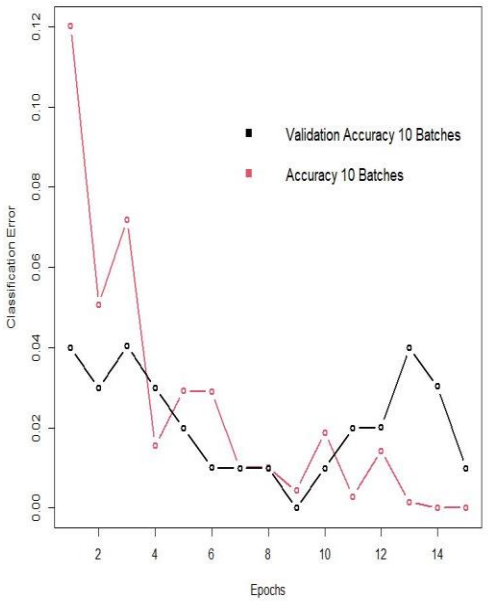
- Loss is the quantitative measure of deviation or difference between the predicted and actual values - it measures the mistakes the CNN makes in predicting the output.
- When loss exceeds validation loss, we have the case of underfitting, a rarity.
- The most common scenario is that of overfitting - i.e., when loss is significantly less than validation loss, as seen in this case, which implies that the model is adapting so well to the training data that it considers random noise as meaningful data. In other words, the model fails to generalize well to previously unseen data.
- The ideal scenario is when loss is approximately equal to validation loss, as that would mean that the model is perfectly fitting on both training and validation data.

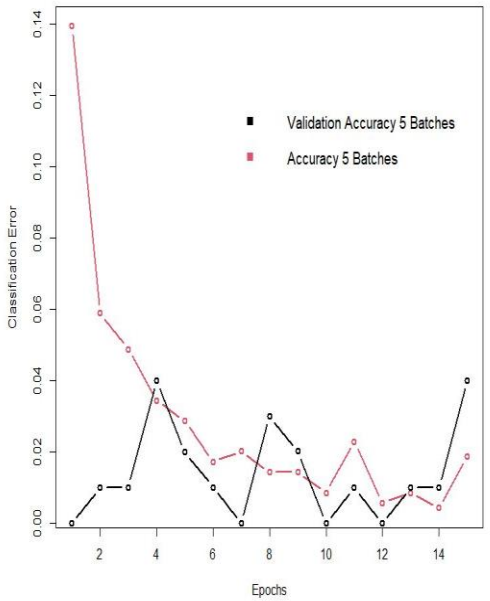
Analyses: CNN Model, Classification Error and Loss (Simple Architecture)

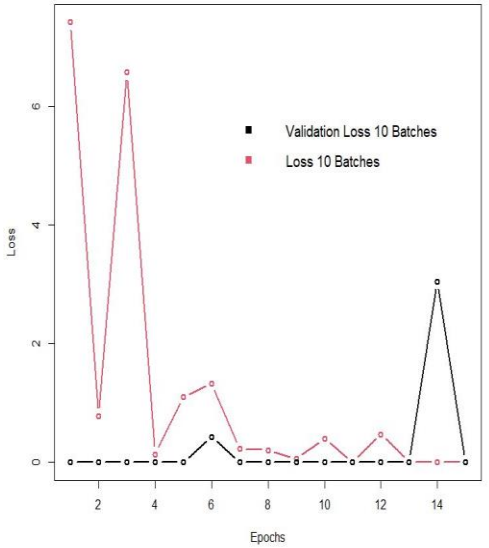
In [9]:
```
datagen = tf.keras.preprocessing.image.ImageDataGenerator(validation_split=0.3)
# load and iterate training dataset
image_width, image_height = 100, 100
trainsamples=700
validsamples=100
testsamples=150
epochs=15
batchsize=5

training = datagen.flow_from_directory("G:/ACADEMIC AND RESEARCH-RELATED/JAPAN-2020/covid-chestxray-dataset-master/modelling", cl
validation = datagen.flow_from_directory("G:/ACADEMIC AND RESEARCH-RELATED/JAPAN-2020/covid-chestxray-dataset-master/modelling",

if K.image_data_format()=='channels_first':
    inputshape=(3, image_width, image_height)
else:
    inputshape=(image_width, image_height, 3)
train_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_f
test_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
if K.image_data_format()=='channels_first':
    inputshape=(3, image_width, image_height)
else:
    inputshape=(image_width, image_height, 3)
train_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_f
test_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
```

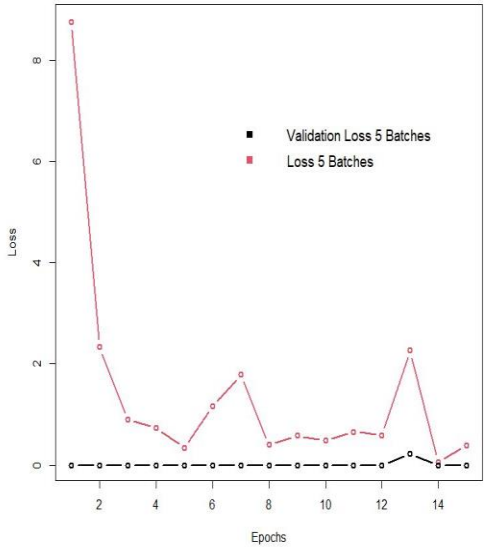
```
Found 651 images belonging to 2 classes.
Found 279 images belonging to 2 classes.
```

In [10]:
```
model=Sequential()
model.add(Convolution2D(32,(3,3),input_shape=inputshape))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Convolution2D(32,(3,3)))
model.add(Activation("relu"))
#model.add(MaxPooling2D(pool_size=(2,2)))
#model.add(Convolution2D(64,(3,3)))
#model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(64))
model.add(Activation("relu"))
model.add(Dropout(0.25))
model.add(Dense(1))
model.add(Activation("sigmoid"))
model.summary()
```









Concluding Remarks: Dealing with Randomness and Concept Drift in Large Datasets

Algorithm 1 SMA-Sample, Measure, Assess

1: **procedure** SMA
2: Set $\mathbf{X} = [x_{i,j}]$: Accessible Data Source
3: Learn $F(\phi) = \underbrace{(P)}_{x,y \sim D} [\phi(x) \neq y]$ based on a chosen learning model
4: Set the number of iterations to a large number K
5: **Initialise:** $\Theta_{tr} := \Theta_{tr}(.)$: Training Parameters
6: **Initialise:** $\Theta_{ts} := \Theta_{ts}(.)$: Testing Parameters
7: **Initialise:** $\Pi_{cp} := \Pi_{cp}(.)$: Comparative Parameters
8: **Initialise:** s as a percentage of $[x_{\nu,\tau}]$, say 1%
9: s_{tr} : Training Sample $[x_{\nu,\tau}] \leftarrow [x_{i,j}]$ extracted from $\mathbf{X} = [x_{i,j}]$
10: s_{ts} : Test Sample $[x_{\bar{\nu},\tau}] \leftarrow [x_{l \neq i,j}]$ extracted from $\mathbf{X} = [x_{i,j}]$
11: **for** $i := 1 \rightarrow K$ **do**: Set K large and iterate in search of optimal values
12: **while** $s \leq 50\%$ of $[x_{\nu,\tau}]$ **do** Vary sample sizes to up to the nearest integer 50% of X
13: **Sampling for Training:** $s_{tr} \leftarrow X$
14: **Sampling for Testing:** $s_{ts} \leftarrow X$
15: **Fit Training and Testing Models** $\hat{\mathcal{L}}_{tr,ts} \propto \Phi(.)_{tr,ts}$ with current parameters
16: **Update Training Parameters:** $\Theta_{tr}(.) \leftarrow \Theta_{tr}$
17: **Update Testing Parameters:** $\Theta_{ts}(.) \leftarrow \Theta_{ts}$
18: **Compare:** $\Phi(.)_{tr}$ with $\Phi(.)_{ts}$: Plotting or otherwise
19: **Update Comparative Parameters:** $\Pi(.)_{cp} \leftarrow \Phi(.)_{tr,ts}$
20: **Assess:** $P(\Psi_{D,POP} \geq \Psi_{B,POP}) = 1 \Longleftrightarrow \mathbb{E}[\Psi_{D,POP} - \Psi_{B,POP}] = \mathbb{E}[\Delta] \geq 0$
21: **end while**
22: **end for**
23: **Output the Best Models** $\hat{\mathcal{L}}_{tr,ts}$ based on $\mathbb{E}[\Delta] \geq 0$
24: **end procedure**

ALLOCATION RULE ERRORS DUE TO DATA RANDOMNESS			
Population	Training	Cross-Validation	Testing
Ψ_{POP}	Ψ_{TRAIN}	Ψ_{XVALID}	Ψ_{TEST}

The quantity $[x_{\nu,\tau}] \leftarrow [x_{i,j}]$ is the training dataset and its initialisation, in step 8, as a percentage of the full data depends on the nature and magnitude of the data source. The same applies to initialisation of $[x_{\nu,\tau}] \leftarrow [x_{l \neq i,j}]$. The loop from step 11through 19 involves sampling through the data with replacement, fitting the model and updating the parameters. The choice for the best performing model is carried out at step 20, where $P(\Psi_{D,POP} \geq \Psi_{B,POP})$ denotes the probability of the population error being greater than the training error.

Conclusion

- This paper highlighted the potentials of combining underlying domain knowledge, data science-technical and soft skills and interdisciplinarity. It also highlighted the impact the COVID-19 pandemic has had on SDGs. Interdisciplinarity is particularly crucial. While CNN models can detect patterns that might go unnoticed to the human eye, for all their power and complexity, they do not provide thorough interpretations of the imagery data. That is why we need the Venn diagram.
- We fulfilled the objectives set, highlighting key challenges and opportunities that researchers face when working with COVID databases (repositories) such as data segmentation, interoperability and free access, which are basically success stories. Going forward, we have the following suggestions for the scientific community.
- There are many lessons we can take from the COVID-19 pandemic, not least how we generate and share data. It is in this context that lessons derived from COVID-19 can help enhance our understanding of the mutual impact–positive and negative, resulting from our interaction with our environment. That is, viewing the bigger picture.
- There can be no better way to view this bigger picture than through the Sustainable Development Goals (SDGs) initiative. SDG monitoring in post COVID–19 conditions should reflect realities in a spatio-temporal context, focusing on, *inter-alia*, citizen science data, machine learning, IoT and mobile applications. We will need an interdisciplinary approach to respond to new challenges and exploit new opportunities in sectors like manufacturing, agriculture, business, health and education-sectors that have been badly hit by the pandemic. Tracking global variations in recovery strategies in various sectors and addressing real-life issues like food security, innovation, productivity etc will be crucial.
- The X-Ray examples used in this paper are bare basics of deep and machine learning methods for biomedical imaging and related clinical data, which academic, biomedical and industry will need to explore further as a way of decreasing diagnostic errors and developing and scaling novel phenotypes to enhance precision in the medical research related fields.

References

1. Rothan, H. A.; Byrareddy, S. N. The epidemiology and pathogenesis of coronavirus disease (COVID-19) outbreak. Journal of Autoimmunity 2020, 109, 102433.
2. SDG, Sustainable Development Goals. 2015; https://www.un.org/sustainabledevelopment/sustainable-development-goals/.
3. Zambrano-Monserrate, M. A.; Ruano, M. A.; Sanchez-Alcalde, L. Indirect effects of COVID-19 on the environment. Science of The Total Environment 2020, 728, 138813.
4. Bartik, A. W.; Bertrand, M.; Cullen, Z.; Glaeser, E. L.; Luca, M.; Stanton, C. The impact of COVID-19 on small business outcomes and expectations. 2020, 117, 17656–17666.
5. Pan, S. L.; Zhang, S. From fighting COVID-19 pandemic to tackling sustainable development goals: An opportunity for responsible information systems research. International Journal of Information Management, 2020, 102196.
6. Wang, C. J.; Ng, C. Y.; Brook, R. H. Response to COVID-19 in Taiwan: Big Data Analytics, New Technology, and Proactive Testing. 2020, 323, 1341–1342.
7. IUCN, In the spirit of nature. 2018; https://www.iucn.org/news/europe/20181/spirit-nature-everything-connected.
8. Mwitondi, K.; Munyakazi, I.; Gatsheni, B. Amenability of the United Nations Sustainable Development Goals to Big Data Modelling. International Workshop on Data Science-Present and Future of Open Data and Open Science, 12-15 Nov 2018, Joint Support Centre for Data Science Research, Mishima Citizens Cultural Hall, Mishima, Shizuoka, Japan 2018.
9. Mwitondi, K.; Munyakazi, I.; Gatsheni, B. An Interdisciplinary Data-Driven Framework for Development Science. DIRISA National Research Data Workshop, CSIR ICC, 19-21 June 2018, Pretoria, RSA 2018.
10. Kharrazi, A. Challenges and Opportunities of Urban Big-data for Sustainable Development. Asia-Pacific Tech Monitor 2017, 34, 17–21.
11. Kruse, C. S.; Goswamy, R.; Raval, Y.; Marawi, S. Challenges and Opportunities of Big Data in Health Care: A Systematic Review. JMIR Medical Informatics 2016, 4, e38.
12. Yan, M.; Haiping, W.; Lizhe, W.; Bormin, H.; Ranjan, R.; Zomaya, A.; Wei, J. Remote sensing big data computing: Challenges and opportunities. Future Generation Computer Systems 2015, 51, 47–60.
13. ECDC, COVID-19 Data. 2020; https://www.ecdc.europa.eu/en/publications-data.
14. ONS, Office for National Statistics. 2020; https://www.ons.gov.uk/.
15. Cohen, J. P.; Morrison, P.; Dao, L. COVID-19 image data collection. arXiv 2003.11597 2020; https://github.com/ieee8023/covid-chestxray-dataset.
16. Kaggle, Chest X-Ray Images (Pneumonia). 2020; https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia.
17. Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics 1980, 36, 193–202.
18. LeCun, Y.; Jackel, L. D.; Boser, B.; Denker, J. S.; Graf, H. P.; Guyon, I.; Henderson, D.; Howard, R. E.; Hubbard, W. Handwritten Digit Recognition: Applications of Neural Net Chips and Automatic Learning. IEEE Communication 1989, 41–46, invited paper.
19. LeCun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; Jackel, L. D. Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation 1989, 1, 541–551.
20. Krizhevsky, A.; Sutskever, I.; Hinton, G. E. In Advances in Neural Information Processing Systems 25; Pereira, F., Burges, C. J. C., Bottou, L., Weinberger, K. Q., Eds.; Curran Associates, Inc., 2012; pp 1097–1105.
21. Impact of the digital divide in the age of COVID-19. Journal of the American Medical Informatics Association 2020, 27, 1147-1148.

Thank You
Any Questions?