# Intelligent Sensor Fusion with Deep Learning Algorithm

Shehroz Bashir Malik
*Department of Electronic Engineering*
*Hochschule Hamm-Lippstadt*
Lippstadt, Germany
shehroz-bashir.malik@stud.hshl.de

***Abstract*—COVID-19 is a deadly disease whose presence can be verified through X-Ray and CT-Scans. In this paper, Sensor Fusion Techniques will be combined with a Deep Learning Models to detect COVID-19**

***Index Terms*—Sensor Fusion, Deep Learning, Computer vision, COIVD-19**

## I. Introduction

COVID-19 also known as SARS-COV-2 is an alarming disease caused by a novel coronavirus. It spreads primarily through water droplets in air discharged by infected humans when they cough, sneeze or speak [1]. These droplets are not capable of travelling far so they require close contact in order to spread. It directly attacks the infected person's lung and simultaneously damages tissues. Air sacs in an infected person start to be filled with fluid which limits their ability to take in oxygen. Eventually, this also leads to scarring and lesions. This damage can be be identified on X-Ray and CT-scans. There are distinct differences between a healthy person's lungs and one with COVID. This damage does not appear immediately. An important assumption is made in this paper that all scans are of patients who have had COVID for a few days.

Being able to effectively identify COVID-19 in patients' X-rays and CT Scans is an important asset to the medical community. This technology is aimed in assisting doctors in making their final decision.

In this paper, Convolutional Neural Networks will be employed to identify aforementioned scans using Sensor Fusion techniques. Different Architectures will be introduced to compare results as well as narrow down on the most accurate model. Methods will be put forward and experimented on to improve the Deep Learning Model.

Section 2 expands on Sensor Fusion as well as the different types of Sensor Fusion. Section 3 introduces Convolutional Neural Networks as well as its uses. Section 4 analyzes various approaches to improve a deep learning model. Section 5 introduces the architectures that will be used in this in paper. Section 6 dives into the dataset as well as any adjustments and pre-processing. To conclude Section 7 states the methodology as well as the results.

## II. Sensor Fusion

Human beings take in inputs from 5 different sensors measuring taste, sight, sound, smell and touch in order to perceive and evaluate the uncertain world in front of us. In order to explore and analyze increasingly difficult environments, more sophisticated approaches are being developed. Methodology and Approaches are being explored to implement and handle inputs from multiple sensors. The efforts are being undertaken for improving the robustness of control systems as well as creating fully autonomous systems.

Sensor Fusion is defined as the combination of sensory data or their respective outputs in order to achieve an output that provides a more realistic view of the scenario being observed. McKee further categorizes data created for the purpose of being fused as Direct and Indirect Fusion [2]. The former relates to compounding sensor data from a set of heterogeneous or homogeneous sensors, soft sensors, and history values of sensor data. The latter relates while indirect fusion uses information sources like a priori knowledge about the environment and human input. Mckee states that the outputs of both direct and indirect fusion can also be combined.

There are certain limitations to using mono-sensor systems that can be addressed by Sensor Fusion. Issues such as Limited Spatial and Temporal Coverage, Imprecision, Sensor Deprivation and most importantly - Uncertainty can be tackled. Uncertainty is one of the main motivations behind pursuing Sensor Fusion.

The old adage goes two heads are better than one. This is very relevant here as one can have multiple sensors observing a single quantity in order to reduce redundancy. If a sensor fails, there will always be another one observing that quantity. This adds robustness and reliability to the system. Additionally, the confidence in the sensor values is reinforced. Ambiguities in data can also be identified and any irregularities related to malfunctioning sensors can be disregarded. Sensors are not always directly in front of their environments making them susceptible to interference. Having multiple sensors increases dimensional of the measurement space thus decreasing the system's susceptibility. This also improves the system's Spatial and Temporal Coverage. Furthermore, Sensor Fusion reduces system complexity as the sensor data is pre-processed and fused before being sent forward [3]. Thus allowing the system to not having to deal with large streams of ambiguous data. Fusion Processes can be split in 3 distinct categories [4]:

1) Low Level Fusion - The Raw Data is not pre-processed

at the sensor level and instead sent directly to the application. Although this requires more bandwidth and increases system complexity, it does allow one to classify data at a very early stage
2) Feature Level Fusion - Features from raw data is extracted during pre-processing. The feature map generated from this is then sent to the fusion module. It achieves the same results as Low Level Fusion while maintaining low system complexity.
3) High Level Fusion - Pre-processing takes place at the sensor level thus enabling modularity and encapsulation of sensor specific details.

Sensors can be configured as a network and eventually fused. Durrant-Whyte categorizes them in the following manner [5]:

1) Complementary - Sensors are independent of each other and observe various differing aspects of the environment. These measurements can then be appended to each other.
2) Competitive - Sensors observe the independent measurements of the same property. This improves fault tolerance and system redundancy
3) Cooperative - Inputs from independent sensors can be combined to achieve an overview that would be not be possible from the individual sensors.

These categories are not absolute as it is possible to develop hybrid systems that embody similar functionalities.

## III. Convolutional Neural Network

Convolutional Neural Networks consists of multiple interconnected computational nodes that are capable of optimizing themselves in order to achieve the most optimal output [6]. These nodes are known as neurons and are heavily inspired by biological process [7]. They receive an input and proceed to perform certain operations on it. The processing of this input is distributed to the hidden layers. These hidden layers are capable of saving decisions made by each prior layer and calculating whether a stochastic change within itself leads to an improvement of the final output or a detriment. A Convolutional Neural Network is predominantly used for pattern recognition in images due to the fact that they require comparatively brief pre-processing.

## IV. Improving your Deep Learning Model

There are a few parameters that can be adjusted in order to better improve a Deep Learning Model. These include the following:

1) Activation function - Depending on one's scenario, there are few activation functions that can be used. For Binary Classification, one can use the Sigmoid Activation Function. For Multiclass Classification, one can use the Softmax Activation Function and for Multilabel Classification, one can use the Sigmoid Activation Function. When dealing with the Activation Function for hidden layers, Multilayer Perceptron and Convolutional Neural Networks use the Rectified Linear Activation Function. Recuurent Neural Networks can use the Tanh and Sigmoid Activation Activation Function. It is possible to use both for RNNs.
2) Batch Size - Batch Size is the amount of training samples your network can access during each iteration. It is possible to have large batch sizes but this will drastically increase computational complexity. However Domninic Masters et al. [8] determined the optimal batch size lies between 2 and 32.
3) Optimizer - Optimizers are used to decrease the loss function. There are various Optimizers each carrying their respective pros and cons. Common ones are Stochastic gradient Descent, Gradient Descent, Mini-Batch Gradient Descent, AdaGrad(Adaptive Gradient Descent) [9], RMS-Prop (Root Mean Square Propagation), AdaDelta [10] and Adam(Adaptive Moment Estimation). Adam is the most popular choice as it it quite fast and converges rapidly although it consumes more resources than the others [11].
4) Learning Rate - As the Deep Learning Model trains, its weights are updated using an optimization algorithm. The amount or step size at which the weights are updated is known as the Learning Rate. The values of a learning rate usually range from 0 to 1 but certain architectures support larger learning rates. Learning rates with large values result in immediate changes while requiring fewer training epochs and vice versa. With a small learning rate, one runs the risk of over-fitting. On the other hand with a large learning rate, there is a possibility of divergence. The optimal strategy is to experiment with various learning rates often starting from small values and incrementally increasing them. The loss can be graphed as a function of the learning rate which will allow you to find the best fit value.

## V. Architecture

### A. Resnet50

He Kaiming et al. introduced the Residual Network Architecture in 2015 [12]. The Resnet50 Architecture consists of 48 Convolution Layers and 1 MaxPool layer as well as 1 Average Pool layer. Creating aggressively deep models was resulting in a high training error. The authors decided to rectify this by introducing shortcut connections that perform identity mappings. These identify mappings did not add any further parameters thus not leading to an increase in computational time. This also enabled the network to continue making progress even if certain layers have not started learning. In addition, they were also able to nullify the vanishing gradient problem that was plaguing a lot of large architectures. Figure 1 describes the architecture of Resnet50 [13].

### B. VGG16

VGGNet is architecture created by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group at Oxford University [14]. The VGG16 variant has 16 convolutional layers while the VGG19 has 19 layers. It is a very simple

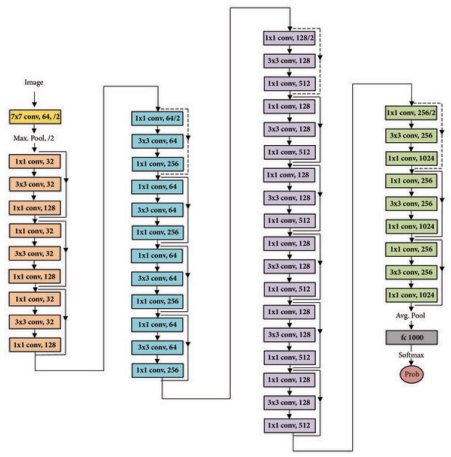

Fig. 1. Block Diagram of the Resnet50 Architecture

architecture that achieves great results when it comes to extracting features from images. Although it has 138 million parameters, the model is already pretrained on the Imagenet Dataset which enables the end user to perform transfer learning on their particular usecase. Figure 2 describes the architecture of VGG-16 [15].

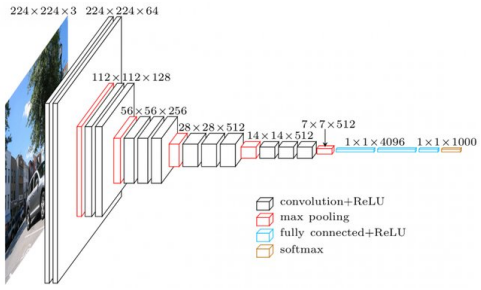


Fig. 2. Block Diagram of the VGG-16 Architecture

## VI. Dataset

The dataset consists of X-ray and CT scans of patient's chests. The original X-ray dataset [16] has approximately 2500 x-ray scans belonging to 2 classes namely: COVID and Non-COVID. On the other hand, the original CT-Scan dataset [17] contains approximately 2500 CT-scans also belonging to 2 classes namely: COVID and Non-COVID. In order to reduce computational complexity and avoid class imbalance, 100 images from all 4 classes were selected at random and fed into the model.

The next step was to implement Data Augmentations in order to further improve the model. Since the dataset is relatively small, Data Augmentations were employed to artificially increase the size by creating realistic versions of the images with slight variations. These variations included rotating, resizing the image, slightly shifting the point of focus and flipping the pictures horizontally. This allows the model to be more cognizant of its objective and increases tolerance to pictures that might not follow the usual norm [18].

Item Transforms were used to apply a Random Resized Crop to the images. Item Transforms applies the augmentation to each image individually. These are then sent to the GPU for processing. Random Resized Crop randomly crops the size and aspect ratio of the image then resizes it to our given size of 224.

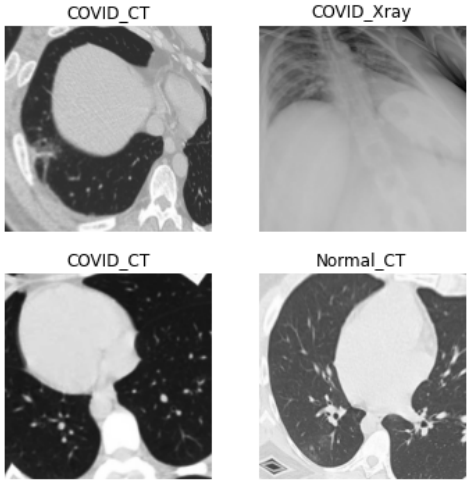


Fig. 3. Dataset with Augmentations

## VII. Learning

The model was run on Ubuntu 20.04 LTS with an i7-8750H and Nvidia GTX-1050 Ti. The Fastai library built for PyTorch was used throughout.

### A. Training with Resnet

The Dataset was initially run through the Resnet50 Architecture with the chosen metric as the error rate. This ran for 10 epochs. This was executed with the sole purpose of determining the optimal learning rate. The learning rate was $2 * 10^{-4}$.

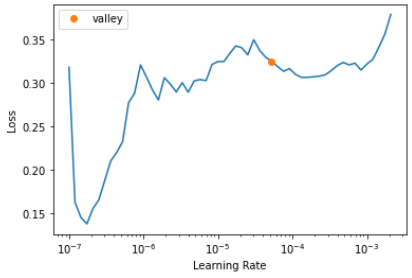


Fig. 4. Learning Rate First Iteration for Resnet50

Caution was taken to not break the pre-trained weights and the other layers. FastAi contains a function called fit one cycle that trains the model at a lower learning rate and incrementally increases it then decreases it in the final section. This allows one to determine the learning rate without

affecting the weights. The error rate decreased as the new learning rate of $1.9 * 10^{-4}$ was used.

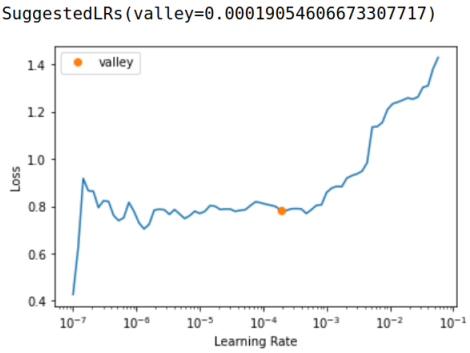


Fig. 5. Learning Rate Second Iteration for Resnet

Although there is a very minor difference between the learning rates, the model certainly improved. The confusion matrix was generated.

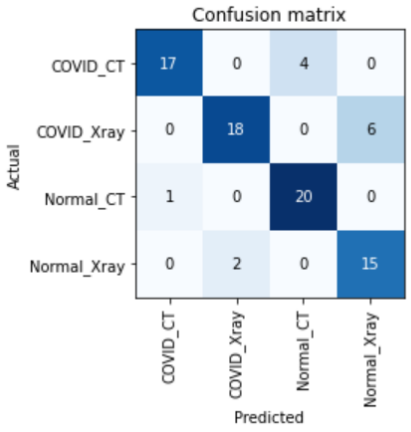


Fig. 6. Confusion Matrix of Resnet50

The Training and Validation loss was also graphed resulting in an accuracy of 84.377%.

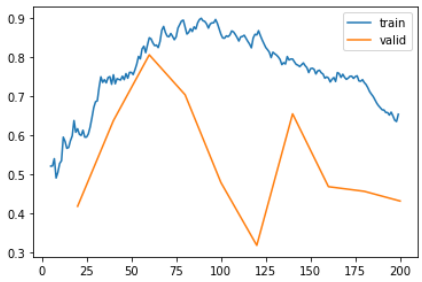


Fig. 7. Training vs Validation Loss of Resnet50

Additionally the Resnet34 architecture was also used, it fared slightly better to its counterpart achieving an accuracy of 85.54%.

### B. Training with VGG

The dataset was run through the VGG16 architecture using the aforementioned methodology. The learning rate was suggested to be $1 * 10^{-4}$. Upon further investigation, the optimal learning rate was $1.202 * 10^{-5}$. The model's accuracy was recorded before optimizing and after. It stayed steady at 92.77% regardless of the changes in learning rate. This did perform better than the previous model.
The model was also run through the other VGG variant. It achieved an accuracy of 90.36%. After finding the learning rate and running it for another 10 epochs, the accuracy dropped to 87.9518

TABLE I
RESULTS

| Architecture | Learning Rate | Accuracy |
|---|---|---|
| VGG19 | Optimal | 90.36% |
| VGG19 | 4.37e-5 | 87.95% |
| VGG19 | 1.58e-6 | 85.54% |
| VGG16 | 1.20e-5 | 92.77% |
| Resnet50 | 2.00e-4 | 84.33% |
| Resnet34 | Optimal | 85.54% |
| Resnet34 | 3.31e-6 | 81.29% |

## VIII. CONCLUSION

In this paper, Sensor Fusion Techniques have been explored. I have succesfully demonstrated ways to implement Convolutional Neural Networks to further augment detection of COVID-19 pushing forward cross-disciplinary techniques. Resnet and VGG Architectures have been investigated. It is safe to conclude that VGG-16 as the best-performing architecture with an accuracy of 92.77%. The other architectures performed in satisfactory manner, all achieving accuracies greater than 81%.

## REFERENCES


[1] F. Wu, S. Zhao, B. Yu, Y. Chen, W. Wang, Z. Song, Y. Hu, Z.-W. Tao, J.-H. Tian, Y. Pei, M.-L. Yuan, Y.-L. Zhang, F.-H. Dai, Y. Liu, Q.-M. Wang, J.-J. Zheng, L. Xu, E. Holmes, and Y.-Z. Zhang, "A new coronavirus associated with human respiratory disease in china," *Nature*, vol. 579, pp. 1–8, 03 2020.

[2] G. T. McKee, "What can be fused?" in *Multisensor Fusion for Computer Vision*, J. K. Aggarwal, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 71–84.

[3] W. Elmenreich and S. Pitzek, "Using sensor fusion in a time-triggered network," in *IECON'01. 27th Annual Conference of the IEEE Industrial Electronics Society (Cat. No. 37243)*, vol. 1. IEEE, 2001, pp. 369–374.

[4] M. Aeberhard and N. Kaempchen, "High-level sensor data fusion architecture for vehicle surround environment perception," in *Proc. 8th Int. Workshop Intell. Transp*, vol. 665, 2011.

[5] H. F. Durrant-Whyte, "Sensor models and multisensor integration," in *Autonomous robot vehicles*. Springer, 1990, pp. 73–89.

[6] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015. [Online]. Available: https://arxiv.org/abs/1511.08458

[7] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193–202, 1980.

[8] D. Masters and C. Luschi, "Revisiting small batch training for deep neural networks," 2018. [Online]. Available: https://arxiv.org/abs/1804.07612

[9] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 07 2011.

[10] M. D. Zeiler, "Adadelta: An adaptive learning rate method," 2012. [Online]. Available: https://arxiv.org/abs/1212.5701
[11] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385
[13] A. Bhatti, M. Umer, S. Adil, M. Ebrahim, D. Nawaz, and F. Ahmed, "Explicit content detection system: An approach towards a safe and ethical environment," *Applied Computational Intelligence and Soft Computing*, vol. 2018, 07 2018.
[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: https://arxiv.org/abs/1409.1556
[15] M. Loukadakis, J. Cano, and M. O'Boyle, "Accelerating deep neural networks on low power heterogeneous architectures," 01 2018.
[16] M. E. H. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M. A. Kadir, Z. B. Mahbub, K. R. Islam, M. S. Khan, A. Iqbal, N. A. Emadi, M. B. I. Reaz, and M. T. Islam, "Can ai help in screening viral and covid-19 pneumonia?" pp. 132 665–132 676, 2020.
[17] E. Soares, P. Angelov, S. Biaso, M. H. Froes, and D. K. Abe, "Sars-cov-2 ct-scan dataset: A large dataset of real patients ct scans for sars-cov-2 identification," *medRxiv*, 2020. [Online]. Available: https://www.medrxiv.org/content/early/2020/05/14/2020.04.24.20078584
[18] J. Howard and S. Gugger, *Deep Learning for Coders with fastai and PyTorch*. O'Reilly Media, 2020. [Online]. Available: https://books.google.de/books?id=yATuDwAAQBAJ