**Real-time Machine Learning Detection of Water Pollution using Convolutional Neural Networks**

Veer Mehta

*Dhirubhai Ambani International School* ***|*** *veeramitmehta2007@gmail.com*

Acknowledgements:

Thank you for the guidance of Jason Jabbour, mentor from Harvard University in the development of this research paper.

***The proceeding paper aims to explore how machine learning techniques leverage real-time monitoring data to provide early warning systems for water pollution events, enabling timely response and intervention. India faces a severe water pollution crisis, impacting public health and ecosystems. This project proposes a machine learning model that analyzes real-time data to create an early warning system for pollution events. This system can empower communities, authorities, and the public to take proactive measures, raising awareness and promoting water resource protection. The project builds upon existing research and datasets, using various forms of data augmentation and a convolutional neural network for detecting the mode of pollution. We introduce a model capable of real-time inference in under 20 ms, leveraging TinyML techniques to quantize our architecture, resulting in a 90% reduction in model size.***





**I. Introduction**

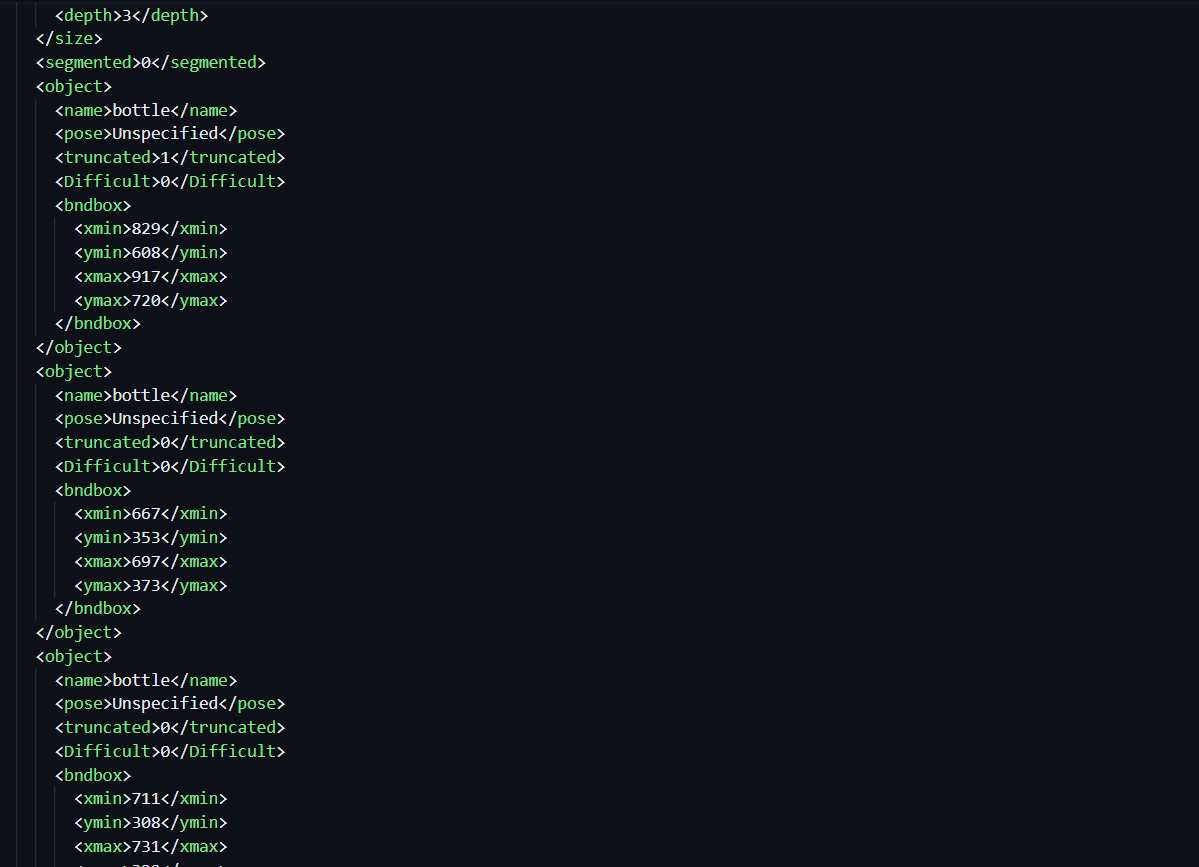
India is grappling with a staggering level of water pollution, with approximatelyhat 70% of its water bodies heavily contaminated (Jennifer 1). This pollution is having a severe impact on public health, with an estimated 1.5 million deaths annually attributed to waterborne diseases (*Pollution causes 2.3 million premature deaths in India, warns report |* 1). Moreover, various ecosystems are under significant stress, with nearly 80% of rivers and lakes facing serious degradation due to pollution. The increase in water pollution has been relentless, with a 55% rise in pollution levels over the last decade alone in India (*CPCB | Central Pollution Control Board* 1).

In response to these statistics, the implementation of a machine learning model offers hope for mitigating the impact of water pollution on the people of India. By analyzing real-time monitoring data, the model can establish an early warning system, helping to detect pollution events and predict potential hazards. This timely intervention will play a crucial role in safeguarding public health and protecting vulnerable ecosystems. By leveraging data-driven insights, this machine learning model has the potential to make a significant difference in combating water pollution and promoting a sustainable and healthy environment for the people of India.

Implementing an early warning system can empower local communities and authorities to take proactive measures. By alerting them to potential pollution events, they can mobilize resources more efficiently and implement preventive measures, reducing the impact of water contamination on their lives. An early warning system can also play a vital role in raising public awareness about water pollution. By sharing alerts and information with the general public, the model can foster a sense of responsibility and encourage individual actions to protect water resources.

In this study, we introduce a machine learning model specifically trained to detect contaminants present on water surfaces. By generating bounding boxes, the model classifies and locates these pollutants. Our model can recognize floating objects in just 20 ms, underscoring its real-time efficacy in monitoring waterways.

**II. METHODOLOGY**

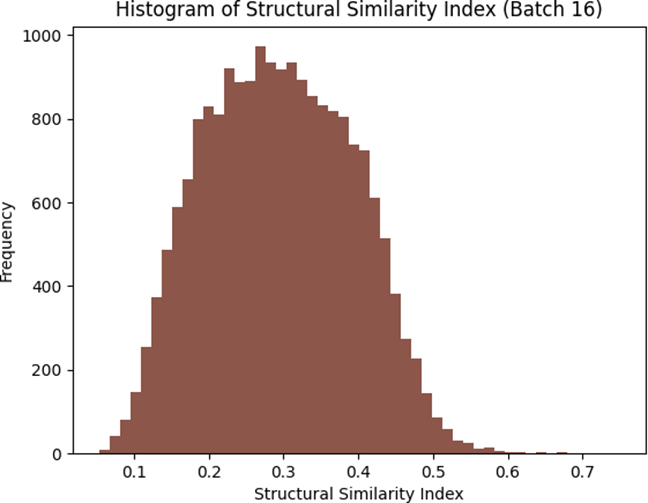
The dataset comprises a total of 2000, 1280 x 720 images, each annotated with corresponding XML file. These XML files provide detailed information regarding the number of pollutants present in each image, as well as their precise location through bounding boxes in the form of (x,y) coordinates. The dataset was sourced from previous research [cite]. A sample image and accompanying XML file looks like the following:



*Image 1: Sample Image Image 2: Sample XML file for data*

In our image recognition project, we are utilizing a Convolutional Neural Network (CNN) due to its effectiveness in handling variations in images. Our approach involves passing the three RGB color channels as inputs to the CNN. To construct our CNN architecture, we incorporate various numbers of Convolutional layers, MaxPooling layers, Dense Layers, and Flatten Layers. Additionally, we conducted tests with different combinations of activation functions, namely rectified linear unit (ReLU), sigmoid, exponential linear unit (ELU), hyperbolic tangent (tanh), and linear activation functions.

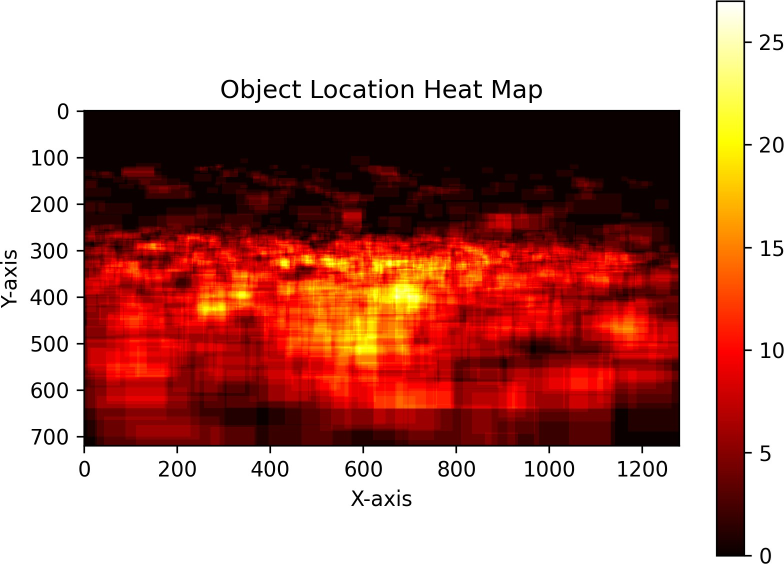
The objective of our neural network is regression-based, aimed at predicting the position of objects within the image. We detect each object using a bounding box methodology, wherein two points in space (x1, y1, x2, y2) are specified, and connecting them outlines the object. To enable our model to detect multiple objects, we will multiply the bounding box dimensions (x1, y1, x2, y2) by the number of objects we want to detect in order to get the number of output nodes of our fully connected neural network portion of our CNN model. This approach enables our model to identify multiple objects simultaneously within a single inference cycle of our image recognition system.

To start getting some intuition behind what our image dataset contains, we gathered the summary statistics of objects found within the images. The average number of objects per image is 2.53 with the minimum being 1 and the maximum being 17. Moreover, we visualize the distribution of the number of objects found within the dataset in the histogram depicted in *Figure 1.*

*Figure 1: Histogram of Structural Sim*

The histogram indicates that most of our images have only a couple of objects. The distribution is skewed right and indicates that most of our images will only have 1 to 3 objects most frequently.

Taking our analysis a step further, we are also interested in discerning the prevalent locations of objects within the images. Understanding the distribution of objects within images is crucial as it enhances the generalizability of a model. This knowledge aids in building a more robust predictive model that can accurately detect pollutants across various scenarios. To achieve this, we extracted the bounding box information from the images and employed this data to generate a heat map. This heat map serves as a visual representation of the frequently occurring object locations within the images.

In *Figure 2*, which illustrates our findings, a clear trend emerges. The majority of the detected objects are concentrated towards the central region of the images. Conversely, there appears to be a notable absence of objects near the upper portion of the images. This observation holds significant implications. It suggests that a model trained solely on these data might struggle to learn the characteristics of objects positioned near the top of the images. Consequently, this insight prompts us to consider strategies for enhancing the model's capacity to accurately predict pollutants situated in less common locations.

*Figure 2*

*Figure 2: Heatmap of prominent pollutant location*

Given that a significant portion of our dataset originates from a single source, we postulate that a substantial resemblance might exist among the images within our collection. This prospect raises the concern that a model trained on such homogeneous data might exhibit limited adaptability when faced with novel data sources or varying environments. To gauge the extent of image similarity quantitatively, we leverage the `sklearn` similarity index package. This tool allows us to compute a similarity index, shedding light on the degree of likeness between images.

The outcomes of our analysis reveal a notable pattern. The majority of images share a similarity index that hovers within the range of 0.3 to 0.5. This range serves as an indicator that our dataset indeed comprises images with substantial similarities. Acknowledging this outcome reinforces the need for careful consideration when training our model, as its performance might be disproportionately influenced by the concentrated similarity within our data. This insight guides our efforts to explore strategies that enhance the model's adaptability across a broader spectrum of scenarios and sources.

**III. RESULTS**

The performance comparison between Convolutional Neural Networks (CNN) and Multi-Layer Perceptrons (MLP) yielded compelling results. The numerical outcomes clearly demonstrate a significant disparity in the model performances, as evidenced by the stark contrast in their respective loss values. After training both models for 20 EPOCS the CNN model reported a 195.6% decrease in the loss compared to the MLP model. These findings indicate a notable advantage of the CNN architecture in effectively addressing the complexities inherent in the real-time monitoring data associated with water pollution events.

**IV. DISCUSSION**

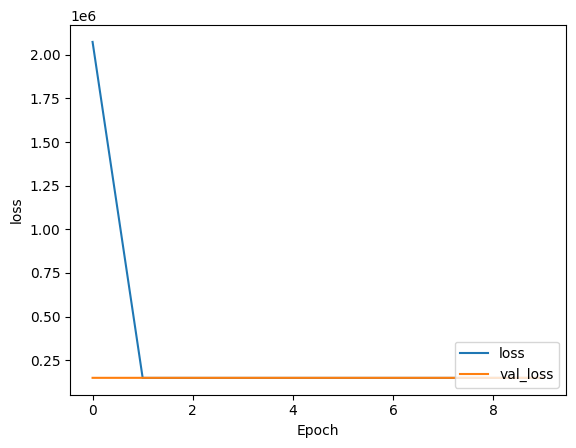
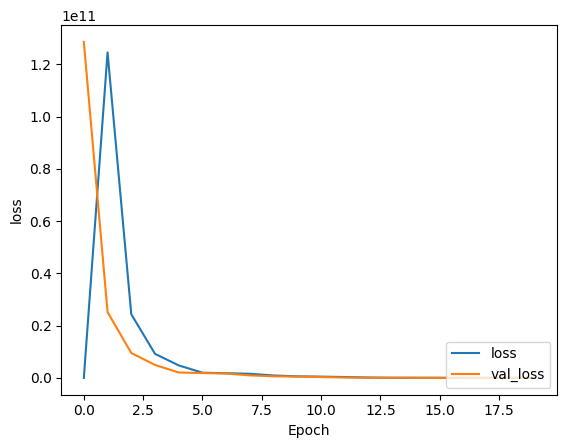
Our investigation initially focused on comparing Multi-Layer Perceptrons (MLPs) and Convolutional Neural Networks (CNNs) in the context of our image recognition project. The comparative analysis provided valuable insights into the performance of these machine learning architectures. The comparison between MLPs and CNNs revealed interesting aspects of their suitability for our task. While MLPs offer flexibility in their architecture, we found that CNNs, which operate at the feature level, were better suited for image data due to their ability to capture spatial relationships.

To gain a deeper understanding, we conducted experiments with various MLP architectures. Through this process, we identified notable relationships among these different architectures.

Notably, we observed that MLP architectures with greater height and width tended to exhibit superior performance. This phenomenon can be attributed to the increased complexity of functions within these architectures. They displayed a heightened ability to approximate the true function mapping our inputs to outputs effectively. Nevertheless, it is crucial to acknowledge the drawbacks associated with MLPs. One significant limitation lies in the flattening process, which results in the loss of spatial information between pixels in the image.

This drawback led us to explore Convolutional Neural Networks (CNNs) further. CNNs offer distinct advantages, as they operate at the feature level instead of the pixel level. This characteristic makes them particularly well-suited for image data.

Our testing with CNN provided for a much lower error rate compared to our MLP model as seen in the following graphs:

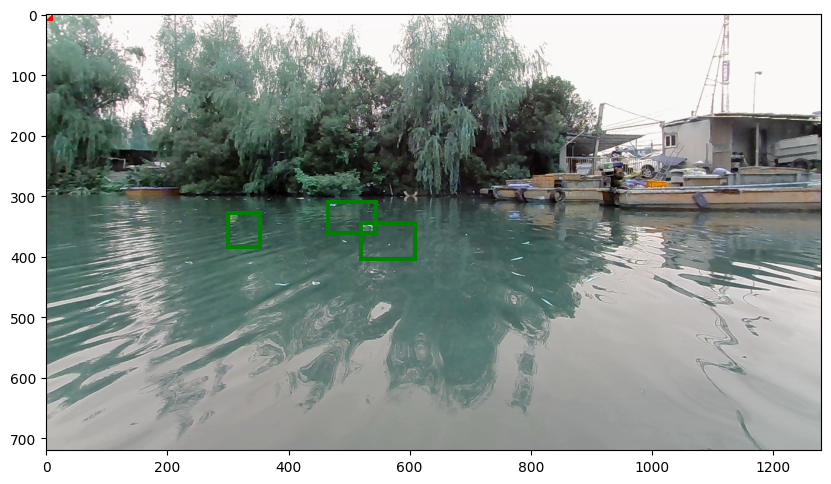


*Figure 3: Error graph forMLP Model Figure 4: Error graph for CNN Model*

As seen the val loss for the MLP Model begins at a loss of 13877020672.0000, whereas for the CNN model begins at a loss of 155520.4219. This is because CNNs are designed to handle spatial hierarchies in data, making them well-suited for tasks like image processing and object detection. They can automatically learn hierarchical features, such as edges, textures, and shapes, which are crucial for understanding objects in images. In the model shown this was accounted for by adding a padding layer to our model’s architecture.

CNNs employ convolutional layers that apply filters to extract local patterns and features from images. These filters capture different aspects of the data, and their shared weights allow the model to recognize these patterns regardless of their location in the image. Moreover, we used 4 convolutional layers with a filter size of 32, as it proved to be the most efficient method of training our model. To address the common challenge of overfitting in deep learning, we also implemented dropout layers in our model. These dropout layers played a crucial role in regularizing the network by preventing it from becoming too reliant on specific features or patterns in the training data. The drop out layers aim to take a first step towards tackling the high similarity of our dataset and the concentration of pollutants towards the center of our image.

Lastly, our CNN included pooling layers that reduce the spatial dimensions of the feature maps while retaining important information. This helps in creating a robust representation of the image's content, which is vital for object detection tasks. For the model shown we used 4 pooling layers with a pool size of (2,2) and a stride of (2,2) as well.

We demonstrate the effectiveness of our model by visualizing the bounding boxes it predicts. As seen in the Figure 5 and Figure 6, the model places bounding boxes around the multiple pollutants floating downstream. 

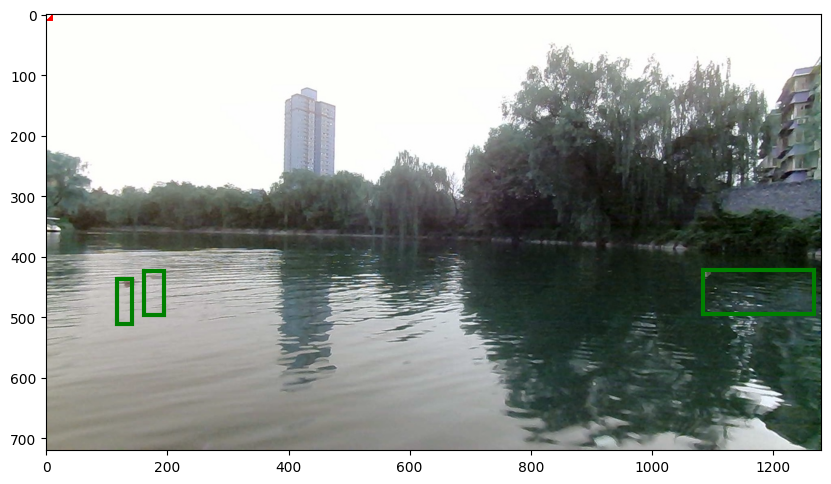
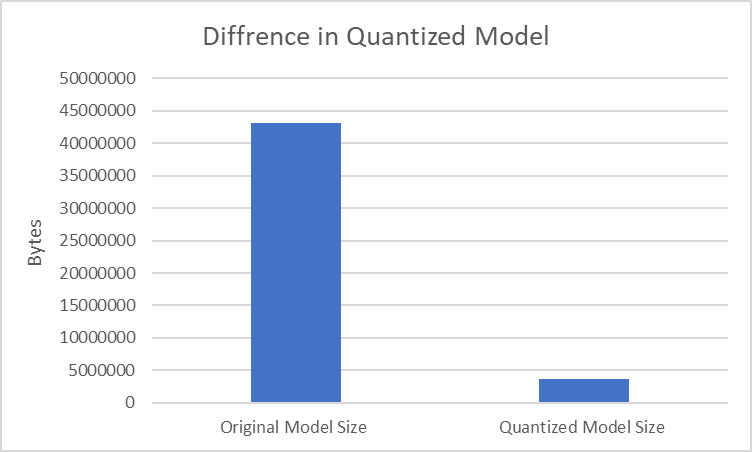


Figure 5: Detected Pollutants Example 1 Figure 6: Detected Pollutants Example 2

Furthermore, the practical deployment of machine learning models for real-time monitoring in remote locations, such as rivers in the context of our project, presents a unique set of challenges. Large-scale models that perform exceptionally well on high-performance computing systems can be impractical for deployment on resource-constrained embedded devices, which are often powered by limited energy sources.

To address this challenge, our research emphasizes the utilization of smaller machine learning models tailored for embedded devices. Specifically, we explore techniques such as quantization, which involves converting the model's weights from 32-bit floating point precision to 8-bit integer precision. This process significantly reduces the model's size, making it well-suited for deployment on power-constrained hardware. As a result, we were able to achieve a remarkable size reduction of 91% of the original model size, as illustrated in the accompanying plot.

*Figure 7: Plot of Original vs Quantized Model Size*

**V. CONCLUSION**

In conclusion, the research undertaken to investigate the utilization of machine learning techniques for early warning systems in water pollution events has yielded valuable insights. Among the models tested, the Convolutional Neural Network (CNN) has emerged as the most effective in leveraging real-time monitoring data for this critical task. Its superior performance in handling spatial features within the data, such as identifying pollutants in images or time-series data, has demonstrated its significance in providing timely alerts for potential water pollution events.

Looking ahead we plan to explore various data augmentation techniques to increase the generalizability of our model. We also plan to explain this model past classifying images of plastic pollutants. We plan to create a model that can identify various types of pollutants such as oil spills, plastics, harmful algae and sewage.

Furthermore, the study's generality can be enhanced by diversifying the types of machine learning techniques employed. In future investigations, we intend to incorporate other algorithms, such as Recurrent Neural Networks (RNNs) for time-series data analysis, Random Forests for ensemble learning, or Support Vector Machines (SVMs) for classification tasks.

All the code behind the project can be found here: <https://github.com/Veer2906/Real-Time-Water-Pollution-Detection.git>

**Works Cited**

*CPCB | Central Pollution Control Board*. https://cpcb.nic.in/nwmp-data/.

Jennifer, Möller-Gulland. *Toxic Water, Toxic Crops: India’s Public Health Time Bomb*. *Toxic Water, Toxic Crops: India’s Public Health Time Bomb*, https://www.circleofblue.org/2018/india/toxic-water-toxic-crops-indias-public-health-time-bomb/.

*Pollution causes 2.3 million premature deaths in India, warns report |*. [*https://www.britsafe.in/publications-and-blogs/safety-management-magazine/safety-management-magazine/2022/pollution-causes-23-million-premature-deaths-in-india-warns-report/#:~:text=Air%20pollution%20(both%20household%20and,)%2C%20closely%20followed%20by%2*](https://www.britsafe.in/publications-and-blogs/safety-management-magazine/safety-management-magazine/2022/pollution-causes-23-million-premature-deaths-in-india-warns-report/#:~:text=Air%20pollution%20(both%20household%20and,)%2C%20closely%20followed%20by%2).

**Extra Info on Figures, Tables and Equations**

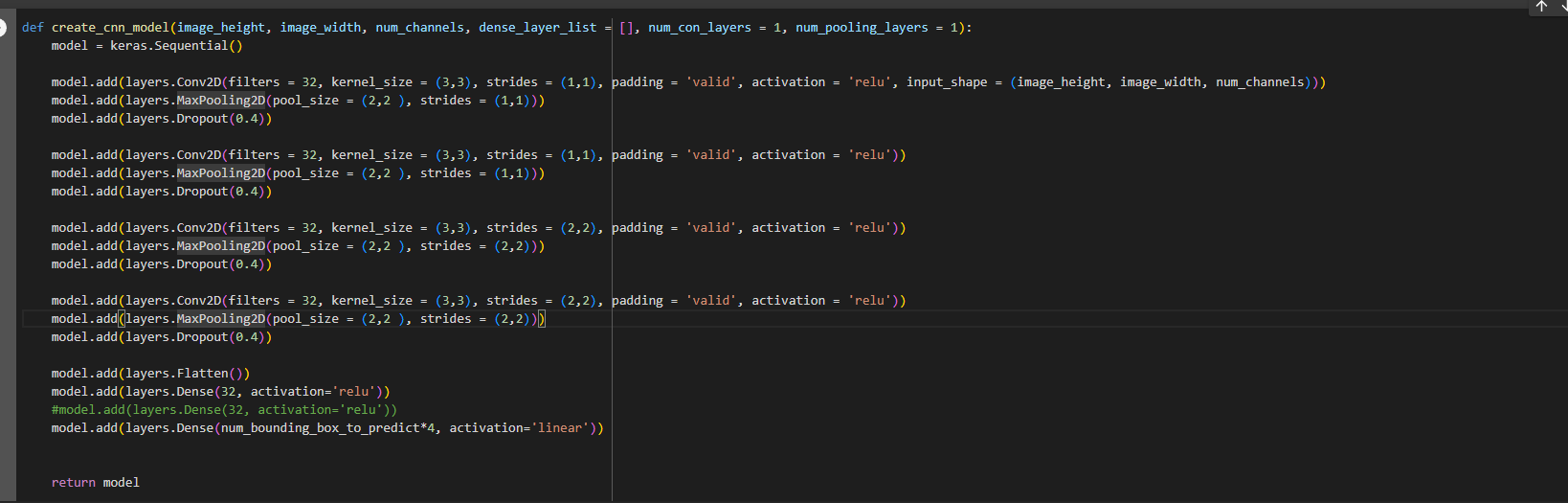


FIGURE 5:Code for Cnn function

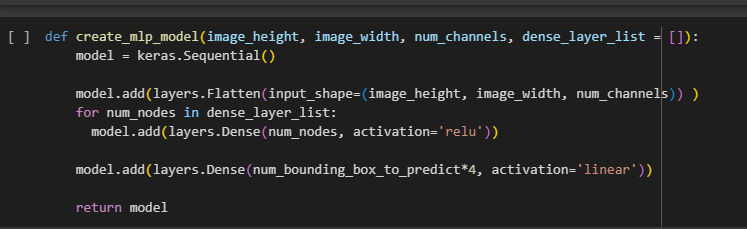


FIGURE 6:Code for MLP function