Journal of Information & Optimization Sciences
ISSN 0252-2667 (Print), ISSN 2169-0103 (Online)
Vol. 47 (2026), No. 7, pp. 2807–2820
DOI : 10.47974/JIOS-2367

Optimizing convolutional neural network architectures : An experimental evaluation of key parameter tuning in smart city building project

Vrushali Rajaram Kadam*
Department of Business Administration
Yashwantrao Mohite Institute of Management
Bharati Vidyapeeth (Deemed to be University)
Karad 415539
Maharashtra
India

Dommaraju Sireesha#
Department of Management
Rushford Business School
Luzern 6010
Switzerland

Abstract

The unregulated and improper monitoring procedures of urban and metropolitan cities lead to poor land usage, uncontrolled urbanization, and pose a risk to the infrastructural tolerability of city life. Effective and continuous computer vision-based artificial intelligence monitoring systems lower the susceptibility risk impacts that can be created in sustainable ecosystems. In this Paper, I am describing my research work on computer vision-based Artificial Intelligence (AI) classification algorithms to monitor and classify the aerial images in monitoring city developments for improved safety, sustainability, and operational efficiency. Computer vision is a branch of AI for interpreting images and videos for software systems. The main objective of this project is to develop a Convolutional Neural Networks (CNN) model for image classification and also optimize parameters to improve the accuracy and reduce the computer resources utilization. The solution includes datasets (aerial images of cities with multiple classes), preprocessing stages, optimizing convolutional layers, preparing feature pyramids, and building neural networks, training networks, and validation

* *ORCID-ID:* 0000-0002-3013-2245
# *ORCID-ID:* 0009-0001-2514-7573

* *E-mail:* vrushali.kadam@bharatividyapeeth.edu (Corresponding Author)
# *E-mail:* sirialgorithms@gmail.com

steps. The results of this project show that increasing neural network layers and the number of training iterations improves accuracy but also consumes significant resources and leads to overfitting; however, reducing the number of neural layers and training iterations results in lower accuracy and underfitting. Therefore, several experiments were conducted to tune parameters and protocols such as convolutional filter dimensions, number of filters, pooling, strides, activation functions, and network size, as well as training-level parameters like learning rate, optimizer selection, epochs, batch size, and weights adjustments. Additionally, a user-friendly interactive dashboard has been built with the Streamlit framework for live demoing of solutions to the end users. It helps users by providing customized statistical reports with analysis of types of constructions and their growth trends. By integrating aerial images, AI, and mapping techniques, the solution supports industrial organizations, government authorities, and environmental agencies in achieving their targets in digitizing city aerial maps and structured planning for initiatives in building smart cities, construction and monitoring.

Subject Classification: *68T05, 68T10, 68T20, 68T40, 68T45.*

Keywords: *Artificial intelligence (AI), Convolutional neural networks (CNN), Smart cities, Image classification, Optimization, Parameter tuning.*

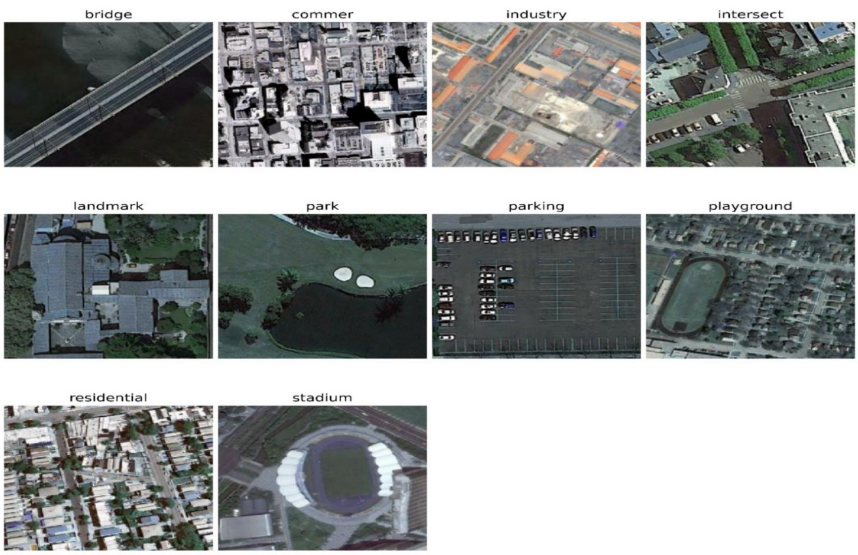
1. Introduction

Computer Vision is a branch of science that systems inference visual data i.e. images/videos. AI systems are capable to interact computational systems and performing tasks without human intervention or minimal intervention. Smart city management is a high investment needed and more manpower and lot of efforts are required to minimize safety risks. The conventional methods are inefficient to collect and classify for precise assessment. Rapid development of AI technology in computer vision, especially Convolutional Neural Networks (CNN) – based classification solutions are serves systems that can detect and classify the visual inspections. The rapid urbanization of the 21st century has led to the emergence of smart cities, which enhance urban ecosystems by integrating AI technologies to improve efficiency, sustainability, and the quality of services provided to organizations and institutions [1, 2]. Computer vision performs various tasks in image classification, segmentation, object detection, tracking and characterization. Deep learning-based image classification serves as the fundamental key for applications to build monitoring systems to categorize visual information into meaningful data [1, 3, 4].

The significant applications are emerging and applications are wide range that includes security and surveillance, optimization of energy

systems and automating the systems control of unauthorized access upon detection of face recognition and suspicious activities and control to the systems [5, 6, 7]. These applications are highly depended on accurate image classification, which transforms raw visual data into momentous classes. The image classification is a fundamental task in computer vision which are process the visual data by extracting hierarchical features (convolutional operations), data acquisition, preprocessing, training algorithm, validations and checking the performance metrics of classification models [2, 5, 6, 7].

This paper demonstrates image classification technologies that are crucial for developing state-of-the-art solutions tailored to specific smart city applications. These solutions an advanced line-up of state-of-the-art image capturing and several parameter tunings also allows it to design the best solution approaches [6, 7, 8]. The main paper objectives are building CNN models, collecting and applying various preprocessing techniques, training AI models, improving the validation protocols and integrate with front-end applications. These AI systems combined with data analytics, sensor data fusion will help to improve productivity, reduce costs and increase better monitoring capabilities [9, 10, 11, 12].



Source: The Figure has plotted in the part of experiment setup. The figure plotted with OpenCv library in python-based environment

Figure 1

Illustrating the Sample Images of Each Class in the Dataset

2. Materials and Procedures

Datasets Information

Data is a crucial component in training the custom CNN-based image classification algorithm. The dataset was collected from Kaggle through numerous validation iterations. While finalizing the dataset, the quality and diversity of this data classes were evaluated in the success of the AI model development. To ensure the high accurate AI model training, sufficient examples of each class had to be collected.

The figure has plotted in the part of experiment setup. The figure plotted with OpenCV library in python-based environment. The overall 7606 images were collected from various types of images that belonged to 10 classes (see Figure 1). The Figure 1 showing the sample image of each class. The dataset comprises a class balance that will give an accurate classification to all categories; it will reduce the model bias towards classes. The classes are Bridge, Commercial, Industrial, Intersection, Landmark, Park, Parking, Playground, Residential and Stadium.

Table 1

Showing the Class Names and Statistics of the Dataset

Class Name	Number of Images	Percentage %
Bridge	800	10.52
Commercial	800	10.52
Industrial	769	10.11
Intersection	800	10.52
Landmark	733	9.64
Park	750	9.86
Parking	800	10.52
Playground	741	9.74
Residential	733	9.64
Stadium	680	8.94

Source: Derived from the original dataset which used in the experiments

Data Aspects and Data Splitting

The following data classes were used as samples for training the CNN classification models. The data splitting is the process of dividing into three parts for given aspect ratios: train dataset (80%), test dataset (10%)

and validation (10%). While splitting the data, random sampling was applied. The validation dataset is used to tune the hyperparameters during the model training process. The randomization of data splitting helps prevent overfitting of the model, assists in hyperparameter tuning, guarantees generalization, and delivers unbiased evaluation. The Python-based split-folders version **(0.5.1)** module is used for splitting the data (see Figure 2) (https://github.com/jfilter/split-folders).

```
#Data split in train, test, and validation
splitfolders.ratio('./Cityscape_Dataset/',output="final", seed=1234, ratio=(0.8,0.1, 0.1))
```

```
Copying files: 7606 files [00:10, 692.59 files/s]
```

Source: The Figure is screen-shot from the coding file for the experiments.

Figure 2

Showing the Python Code for Splitting Image Dataset in Train, Test and Validation.

Data Preprocessing

Widely used image formats are considered. Most of the examined images are in Joint Photographic Experts Group (JPEG/JPG) and Portable

Table 2

Showing the Model Parameters and Preprocessing Steps

Preprocessing step	Parameters	Comment
Rescale	1./255	Pixel normalization
Shear range	0.3	
Zoom range	0.3	
Horizontal Flip and Vertical Flip	True	The image has been flipped in horizontally and vertically
Interpolation	'nearest'	Interpolated images with nearby pixels
class mode	'categorical'	The model needs to classify the 10 classes, hence avoiding binary
target size	(128, 128)	Conducted several experiments with the different sizes. (64, 64) for experiment-1, (128, 128) for experiment-2 and (256, 256) for experiment-3.

Source: The Table 2 numbers are parameter setting for preprocessing of the image. These parameters are hyper-tuned for better performance.

Network Graphics (PNG) formats format. Lightweight JPEG images have been chosen because they are ideal for large datasets, though they are lossy in quality. Several Python-based image preprocessing libraries, such as OpenCV, Pillow, scikit-image, NumPy, and SciPy have been evaluated, and the OpenCV Python library was selected for image processing. Additionally, multiple Python packages will be used to analyze images and videos. The set of preprocessing parameters has been set, and created pipeline has been created for model training with the TensorFlow framework (ImageDataGenerator class). The table below shows the preprocessing parameters used for the training, test and validation datasets. Table 2 describing the preprocessing parameters which applied to the dataset.

Computer Vision-based AI models (Deep learning models)

AI-based advanced CNNs have been developed to classify the images into their corresponding class. The labelled images have been used to train a CNN model. The machine learning model framework has been built through the TensorFlow framework. Several convolutional layers, deep neural networks and an output layer have been added sequential manner with activation functions. The model framework has been initialized with a TensorFlow sequential layer. The first convolutional layer (Conv2D) layer added with filters with 32, kernels with 3X3, activation function of Rectified Linear Unit (ReLU). ReLU is a generally used activation function in deep learning to introduce non-linearity into neural networks, enabling

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_2 (Conv2D)	(None, 126, 126, 32)	896
max_pooling2d_2 (MaxPooling 2D)	(None, 63, 63, 32)	0
conv2d_3 (Conv2D)	(None, 61, 61, 32)	9248
max_pooling2d_3 (MaxPooling 2D)	(None, 30, 30, 32)	0
flatten_1 (Flatten)	(None, 28800)	0
dense_2 (Dense)	(None, 64)	1843264
dense_3 (Dense)	(None, 10)	650

Total params: 1,854,058
Trainable params: 1,854,058
Non-trainable params: 0

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 254, 254, 32)	896
max_pooling2d (MaxPooling2D)	(None, 127, 127, 32)	0
conv2d_1 (Conv2D)	(None, 125, 125, 32)	9248
max_pooling2d_1 (MaxPooling 2D)	(None, 62, 62, 32)	0
flatten (Flatten)	(None, 123008)	0
dense (Dense)	(None, 64)	7872576
dense_1 (Dense)	(None, 10)	650

Total params: 7,883,370
Trainable params: 7,883,370
Non-trainable params: 0

Source: Figure 3 are screenshots of developed model summary for convolutional layers 126, 126 and 254 and 254 layers. The screenshot captured from my experimental procedure

Figure 3

Showing the Model Weights and Deep Neural Networks Layers

them to learn complex patterns. The convolutional layer comprises pooling layers and is built with deep neural networks with 64 units. The deep neural network units were optimized with 64, 128 and 256 units. The output layer added with 10 units to classify the problem into one of the 10 classes based on the probability score. The model is compiled by adding an Adam optimizer. The model summary with layer information shown in Figure 3.

2.5 *Model Training and Validation*

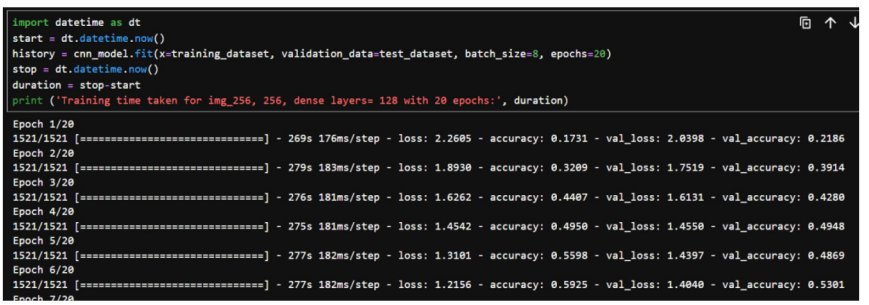
The detection model was trained using labeled datasets, emphasizing the predefined classes of objects. Regular model validation was conducted to ensure accuracy. The following training and validation parameters were used: The AI engine comes with pre-trained neural network on collected images from the selected sources. These pre-trained have been initialized and models further fine-tuned and retrained with images or frames captured by Drone. There shall be multiple iterations until the desired AI model performance to detect waste objects is achieved. A typical iteration consists of drone operation to capture data, cleanse and organize data for AI model finetuning (training and testing), AI model optimization, drone operation to capture images to validate the model, baselines and benchmarked AI model performance and set for goals for next iteration. The first two iterations were performed over script-based AI model optimization, whereas the last two iterations were performed over UI (User Interface)-based AI model optimization. The dataset has been split into train, test, and validation in 0.8:0.1:0.1 ratios. The stratified split strategy ensures a balanced distribution of classes. Metrics are considered to mean average precision, rotated bounding box Intersection of Union (IOU) set for 50%, minimum score: 30%. The various hyperparameters have been evaluated and done best experimentations, based on experimentation observed best parameters are learning rate: 0.03 with warmup and stepped Decay, batch size is 10, loss functions are cross-entropy, smooth L1 for box regression.

Front-end Web Application

The classification model is integrated with a front-end web application to demonstrate the results to the end user. The front-end web application is developed with a Python-based web framework, i.e. Streamlit. The application has been running on the local host, it will be deployed on the server. The front-end web app enables the user interface to upload data and run the AI inferences.

3. Results and Discussion

The experimental methodology applied for the project was systematic, multidimensional and multi-faceted. The research approach commenced with the integrated with the front-end visualization platform which was built in Streamlit framework. This solution ensuring scalability and robustness for future image classification. Data acquisition is a critical stage that unnamed vehicles, Closed-Circuit Television (CCTV) missions were conducted across predetermined areas, capturing diverse visuals. Upon data acquired, the data underwent predefined preprocessing steps pipeline to identify and categorize the images. These preprocessed images utilized to train the computer vision based CNNs that augmentation their image classification abilities. Subsequently, the results are pushed in to a wide-ranging front-end platform through Streamlit. The statistics and analytics processed the test images, and the results were streamlined and produced insights. The whole mechanism and methodology was reinforced by constant validation, manual feedback, and optimization loops, safeguarding the project's accuracy and relevance. The below Figure 4 provides the code snippet of parameters and training procedure of AI models.



```
import datetime as dt
start = dt.datetime.now()
history = cnn_model.fit(x=training_dataset, validation_data=test_dataset, batch_size=8, epochs=20)
stop = dt.datetime.now()
duration = stop-start
print ('Training time taken for img_256, 256, dense layers= 128 with 20 epochs:', duration)
```

```
Epoch 1/20
1521/1521 [==============================] - 269s 176ms/step - loss: 2.2605 - accuracy: 0.1731 - val_loss: 2.0398 - val_accuracy: 0.2186
Epoch 2/20
1521/1521 [==============================] - 279s 183ms/step - loss: 1.8930 - accuracy: 0.3209 - val_loss: 1.7519 - val_accuracy: 0.3914
Epoch 3/20
1521/1521 [==============================] - 276s 181ms/step - loss: 1.6262 - accuracy: 0.4407 - val_loss: 1.6131 - val_accuracy: 0.4280
Epoch 4/20
1521/1521 [==============================] - 275s 181ms/step - loss: 1.4542 - accuracy: 0.4950 - val_loss: 1.4550 - val_accuracy: 0.4948
Epoch 5/20
1521/1521 [==============================] - 277s 182ms/step - loss: 1.3101 - accuracy: 0.5598 - val_loss: 1.4397 - val_accuracy: 0.4869
Epoch 6/20
1521/1521 [==============================] - 277s 182ms/step - loss: 1.2156 - accuracy: 0.5925 - val_loss: 1.4040 - val_accuracy: 0.5301
Epoch 7/20
```

Source: Figure captured from the original experiment's execution

Figure 4

Showing the Code Snippet of Training the Classification Algorithm

Performance Metrics

The trained AI models proven a high accuracy rate achieving maximum performance for the 10% of the dedicated test data provided. The commonly used metrics are in computer vision are accuracy score. precision and recall. The other metrics like precision, recall, and f1 measures have been considered. The dataset composed with balance the

classes (see the reference Table in section 2.1 and Table 1). However, the accuracy score is measured on the level and overall measure. The accuracy score refers to the accuracy of AI models in classifying of interest classes of images which we used in training process of machine learning models. The below table showing the accuracy score against to each class compared between the CNN model with image size 256 X 256, convolutional layers 128 and Residual Network (ResNet50) layered model. The accuracy score calculated total of true positives and true negatives divided by the total number of test images passed to corresponding class. The CNN models with image size 256 and hidden layers units 128 layers increased the training time and larger extension of features. However, it showing less accuracy score compared with ResNet50 model. The Resnet models significantly increases the model accuracy. It even reaches more than 70% of overall accuracy, its almost 7 folds higher than the CNN models. The corresponding numbers for these ratios shown in the confusion matrix heat maps in the below section.

Table 3

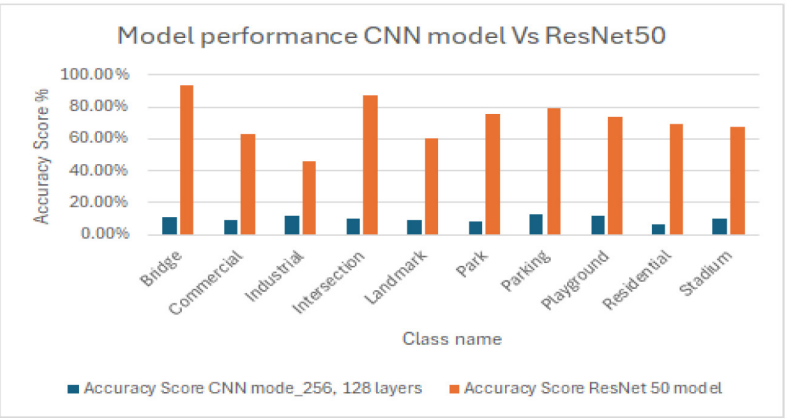
Performance Score of CNN Models and Resnet Models

S. No.	Class Name (Image)	Accuracy Score CNN mode_256, 128 layers	Accuracy Score ResNet 50 model
1	Bridge	11.25 %	93.5 %
2	Commercial	8.75 %	62.79 %
3	Industrial	11.53 %	45.71 %
4	Intersection	10.01 %	87.01 %
5	Landmark	9.45 %	60.8 %
6	Park	8.00 %	75.36 %
7	Parking	12.5 %	79.54 %
8	Playground	12.0 %	73.43 %
9	Residential	6.75 %	69.33%
10	Stadium	10.29 %	67.12%

Source: The relative performances from the model results.

The below Figure 5 shows the model accuracy score of the model performance compared between the CNN built model with ResNet50 model. The results evidencing that more convolutional operations are improving the significance accuracy score. For each class accuracy scores are increased almost 7 times than used of CNN model based over the

ResNet50 (see Table 3). During the model training the number of epochs used same as 20 epochs time. The ResNet model utilized the more computational power than CNN model since more of the convolutional operations in ResNet50 Models.



Source: The Figure 5 plotted with original numbers of the model performance. The graph plotted in excel

Figure 5

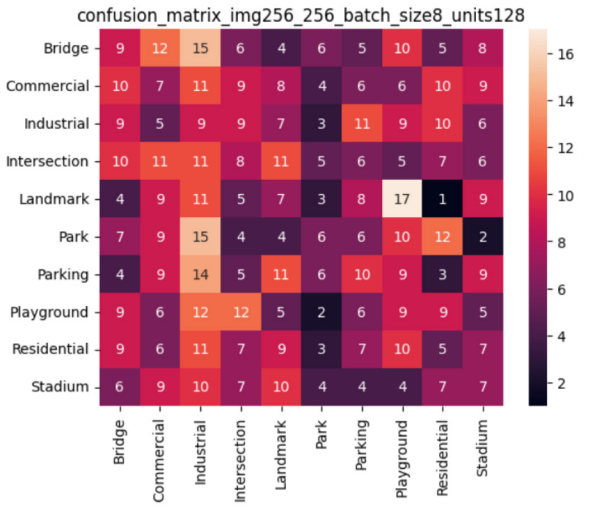
Showing the Model Performance Towards the Classes

The ReeNet50 model took almost 180 minutes to complete 20 epochs on the other hand CNN model training process has finished the 90 minutes for the same number of epochs and almost the same image size. Its proving that ResNet50 models are more efficient in accuracy point of view, however, the model weights are heavy as well more computation power also needed. We obtained a high accuracy score in bridges classification, since bridges structure is simple and spatial data distribution is simple compared with other class images. The minimal accuracy score obtained in industry class, only 45% obtained. The main reason behind the industrial class images showing the similarity behavior (pixel wise) with parking areas. The probability of finding two instances between industries and parking areas and in presence of more images becomes lesser accurate than others. We have implemented a binomial distribution to calculate such probabilities. By evidencing these probability calculations, proved insights into the probabilities associated with finding classes of different sizes and spatial distributed images in more efficiently. Hence AI model has done several

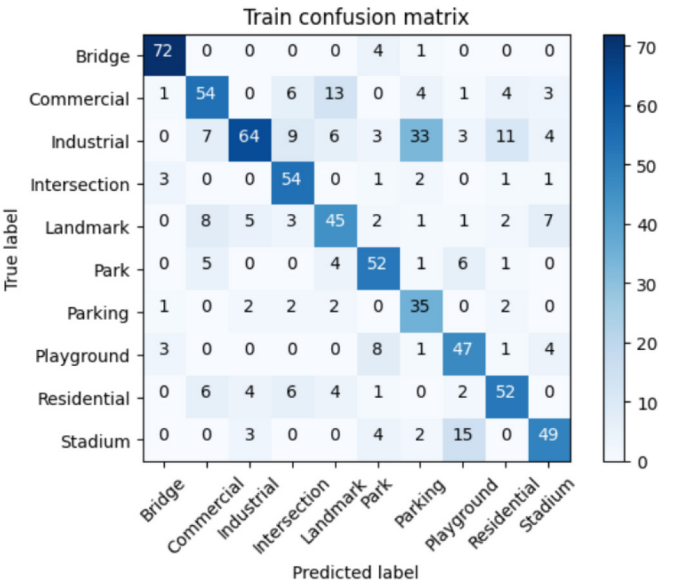
misclassifications during the testing phase. So, we conclude that more images of industries are needed to update AI model weights.

Confusion Matrix

The confusion matrix is a performance table illustrates that all visualization of the classification report of the machine learning algorithm specifically when the use cases are classification, however if use case is a regression problem the we Use Root Mean Squared Error (RMSE) values considered rather than confusion matrix. The confusion matrix delivers AI model test sample results insights into the number of images are correctly classified that is True Positive (TP), the number of images are misclassified False Negatives (FN), and how often images are mistaken i.e. False Positive (FP). The confusion matrix provides not only the accuracy of the model but also provides the how many of the classes or instances are showing the model errors. The rows signify the real image classes, and the columns signify the computer vision-based AU model predictions. The below image (see Figure 6) showing the confusion matrix of the two different computer vision models. The left image is the reswhich built with 128 dense layers and followed image size with 256X256. The right-side image is a result of the ResNet 50 model performance. Both models were trained on same Central Processing Unit (CPU) machine with following same number of epochs as well batch size. The confusion matrices are shown below:



Contd...



Source: The numbers captured from the model accuracy classification towards 10 classes. The matrix has plotted with the original numbers from model matrix. The graph plotted in python based matplotlib. pyplot library

Figure 6

The Confusion Matrix with The Relative Values

Front-end Web Application

To showcase and interact with the results a python-based web application was developed with the Streamlit framework. The web application has been developed for integrating the trained AI models and required pipeline is being built by adding necessary libraries and their versions in the maintaining virtual environments. The front-end application designed with user-friendliness features that offers a wide range of users. Users can smoothly upload images in all formats to the system; it is streamlined process safeguards that the data is readily available for analysis and modeling by the AI algorithms classify the class segment. The web application summarizes the analysis statics and analytics to view data in a visual appearance that provides the overall understanding of city trends and patterns from the AI model inferences. It will enhance the smart city authorities to do effective decision-making.

4. Conclusions

The computer vision based advanced AI application has successfully built for process and classify the visual basic images. The application was integrated with frontend application which was built with python-based framework that Streamlit. The various experiments have been set and evaluated the parameters and dense layers, image size and batch sizes. The AI solution have been validated during the training time as well successfully tested with unseen data. The test inferences increasing dense layers and convolutional layers consume the high computational resources as well provide high accurate in classifying the designated classes. The results shown convolutional operations are crucial to extract the features from the image dataset. The high and low accuracy has been observed in bridge and industrial classes respectively. The synergy between Machine Learning (ML) classification and vision-based technology and has potential applications far beyond the scope of the smart city building projects.

References

[1] S. Niu and J. Zhang, "Image processing based on convolution neural network," *Electronics,* vol. 14, no. 23, Art. no. 4649 (Nov. 2025), doi: 10.3390/electronics14234649.

[2] Y. Akbari, N. Almaadeed, S. Al-Maadeed, and O. Elharrouss, "Applications, databases and open computer vision research from drone videos and images: A survey," *Artificial Intelligence Review,* vol. 54, pp. 3887–3938 (Feb. 2021), doi: 10.1007/s10462-020-09943-1.

[3] V. Vasan, N. V. Sridharan, R. J. Balasundaram, and S. Vaithiyanathan, "Ensemble-based deep learning model for welding defect detection and classification," *Engineering Applications of Artificial Intelligence,* vol. 136, Art. no. 108961 (Oct. 2024), doi: 10.1016/j.engappai.2024.108961.

[4] Z. Niu, H. Pi, D. Jing, and D. Liu, "PII-GCNet: Lightweight Multi-Modal CNN Network for Efficient Crowd Counting and Localization in UAV RGB-T Images," *Electronics*, vol. 13, no. 21, Art. no. 4298 (2024), doi: 10.3390/electronics13214298.

[5] G. K. K. Leizer, "Possible areas of application of drones in waste management during rail accidents and disasters," *Interdisciplinary Description of Complex Systems*, vol. 16, no. 3-A, pp. 360–368 (Sep. 2018).

[6] M.-R. Hsieh, Y.-L. Lin, and W. H. Hsu, "Drone-based object counting by spatially regularized regional proposal network," in *Proc. 2017 IEEE Int. Conf. Computer Vision (ICCV)*, pp. 4165-4173 (2017), doi: 10.48550/arXiv.1707.05972.

[7] Y. Chen, J. Wang, and G. Wang, "Intelligent welding defect detection model on improved R-CNN," *IETE Journal of Research*, vol. 69, no. 12, pp. 9235-9244 (Mar. 2022), doi: 10.1080/03772063.2022.2040387.

[8] D. Ke, "AI-driven closed-loop extrusion-based bioprinting system: From dynamic regulation to medical translation," *Bioprinting*, vol. 54, Art. no. e00459 (Apr. 2026), doi: 10.1016/j.bprint.2025.e00459.

[9] Y. Li, B. Yu, B. Wang, T. H. Lee, and M. Banu, "Online quality inspection of ultrasonic composite welding by combining artificial intelligence technologies with welding process signatures," *Materials & Design,* vol. 194, Art. no. 108912 (Sep. 2020), doi: 10.1016/j.matdes.2020.108912.

[10] S. Perri, F. Spagnolo, F. Frustaci, and P. Corsonello, "Welding defects classification through a convolutional neural network," *Manufacturing Letters*, vol. 35, pp. 29–32 (Dec. 2022), doi: 10.1016/j.mfglet.2022.11.006.

[11] K. V. Krishna Kishore, V. R. R. Chirra, and U. Srinivasulu Reddy, "Optimized stacked deep ensemble classifier using hyperparameter tuning for facial expression recognition," *Journal of Information and Optimization Sciences*, vol. 46, no. 6, pp. 2005–2015 (Oct. 2025), doi: 10.47974/JIOS-2029.

[12] G. Abdelhady, A. Mohsen, and N. Abdelnaser, "Hybrid CNN models for suspect facial recognition system," *Journal of Information and Optimization Sciences*, vol. 47, no. 2, pp. 539–551 (Feb. 2025), doi: 10.47974/JIOS-1606.

Received December, 2025