ISSN: 3030-3346

# DEVELOPMENT OF A MULTI-CLASS MODEL FOR CLASSIFICATION OF LAND, AIR AND WATER TRANSPORT BASED ON THE RESNET34 ARCHITECTURE.

*B.B.Muminov[1], A.Yu.Dauletov[2], N.Sh.Matyakubova[3]*

*[1]Tashkent State University of Economics, Department of Artificial Intelligence,*
*[2]Alfraganus University, Department of digital technologies*
*[3]Tashkent State University of Uzbek Language and Literature named after Alisher Navoi*

*K E Y W O R D S*

object detection, vehicle detection, multi-class classification, CNN, air, land, water.

*A B S T R A C T*

This paper focuses on the identification and classification of land, air and water vehicles. The study used a deep learning approach to identify the type of vehicles based on images. In particular, a multi-class vehicle detection model was developed using a deep convolutional neural network based on the ResNet-34 architecture on a specially prepared dataset. The use of these technologies is becoming important not only for tracking and monitoring moving objects, but also for reducing traffic congestion, preventing traffic accidents and improving the reliability of unmanned vehicles.

## Introduction

Today, the development of artificial intelligence and deep learning technologies is becoming relevant in our republic and around the world. Especially in the fields of medicine, transport and industry, the possibilities for optimizing processes and increasing efficiency are expanding through the use of deep neural networks. Deep networks of the ResNet-34 model provide high accuracy and efficiency of image analysis and object detection, which makes this model widely used not only in scientific research, but also in practice. Therefore, the development of this direction makes a great contribution not only to technological, but also to economic and social development.

In the field of deep learning and convolutional neural networks (CNNs), increasing the depth of the network is important for improving efficiency and accuracy. But as networks get deeper, gradient loss during training becomes a problem. These problems reduce the performance of deep networks and make the training process more difficult. This is where the ResNet architecture comes into play, particularly the ResNet-34 model's approach to solving the gradient loss problem using residual connections.

ResNet-34 (Residual Network 34-layers) is a deep convolutional neural network designed for image classification. This model consists of 34 layers and helps to effectively train deep networks through residual connections. Residual connections ensure that each layer of the model has a positive impact on the training process, speeding up training and improving accuracy. ResNet-34 allows you to classify images with high accuracy. In this paper, the ResNet-34 model is trained on the Open Images v4 (OIDv4 ToolKit) dataset, which contains 1.7 million images and over 600 classes.

## Literature Review

The use of machine learning and artificial neural networks has increased in recent years [1]. Together with image processing techniques, these applications are used to build image classification and object detection models and applications [2]. Such applications are used in all areas, from business to cybersecurity [3,4]. A similar trend has been observed in models and applications developed specifically for vehicle detection. Datasets used for vehicle detection usually focus on only one type of domain (e.g. land, air, sea) or a specific type of vehicle [5,6,7]. One of the most famous examples in this area is license plate recognition applications [8]. K. Avezov and B. Mominov investigated a method for detecting and alerting fire in ship compartments using deep learning and computer vision methods [9]. Shao and Wang built a saliency-aware CNN in YOLO version 2 to build a ship detection model to aid in coastal monitoring [10]. They used an early version of the SeaShips image dataset. Their model achieved 0.874 mAd. Arabi and Hagitat [11] proposed a single-sensor detector (SSD) model with the MobileNet base network to detect construction equipment and assist in construction site management. The proposed model used the MobileNet base network for feature extraction and Jaccard overlay to match objects in the SSD image. The model achieved an accuracy of over 91%. Sang and Wu [12] used a YOLO version 2-based model to detect ground vehicles. The model was trained and tested on the Beijing Institute of Technology (BIT)-Vehicle and CompCars datasets.

A review of the literature on vehicle detection shows that most of them focus either on specific domains (e.g. air, land, or sea) or specific vehicle types [13,14,15]. In addition, most of them use widely known publicly available datasets to train and test models. These datasets are either created specifically for a specific vehicle type or are used to identify a general object with vehicle images. Our study aims to develop a model for fast and high-accuracy vehicle detection in all domains by overcoming the above-mentioned shortcomings.

## Multi-class classification

Multi-class classification is a machine learning task that aims to classify samples into one of three or more classes. Deep learning methods for multi-class classification are particularly effective for image-based tasks and outperform traditional machine learning methods. Deep learning models outperform traditional machine learning algorithms that require manual feature extraction because they are able to detect local and global features using a multi-layer architecture. This improves accuracy and generalization in multi-class classification tasks [16,17,18].

## Convolutional Neural Network (CNN)

CNNs are the most widely used deep neural networks that are used in areas such as image and video recognition [6]. The ResNet-34 model can be expressed as follows:

**Input Image:** In this step, the input image is fed to the network. The image shows an image of a car, which is the input to be processed by the network in the subsequent steps.

**Convolutional Layers:** ResNet-34 also consists of base convolutional layers that extract features from the image. It is combined with convolutional layers (34 layers in total) and residual connections as shown in the figure.

**Binding Layers:** In ResNet-34, pooling layers are used to reduce the size of feature maps and improve the overall efficiency of the model. These layers are important for compressing the image data while preserving the features.

**Fully Connected Layer:** This layer is responsible for determining the class of the image. The ResNet-34 model also has a fully connected layer as the last layer, which is used to classify the class of the image based on the training data.

**Residual Connections:** The main feature of the ResNet-34 model is that it uses residual connections. These connections help make the training at the deep layers of the model more efficient and solve problems like gradient loss.

**Predicted Class:** After creating a fully connected layer on the image, the model predicts a class for the input image. In the ResNet-34 model, at this stage, an image that belongs to one of the 600 classes is classified as the corresponding class. These classes consist of various objects from the ImageNet dataset, such as phones, vehicles, etc. This predicted class is the final output of the model and represents the most likely class that belongs to the image.

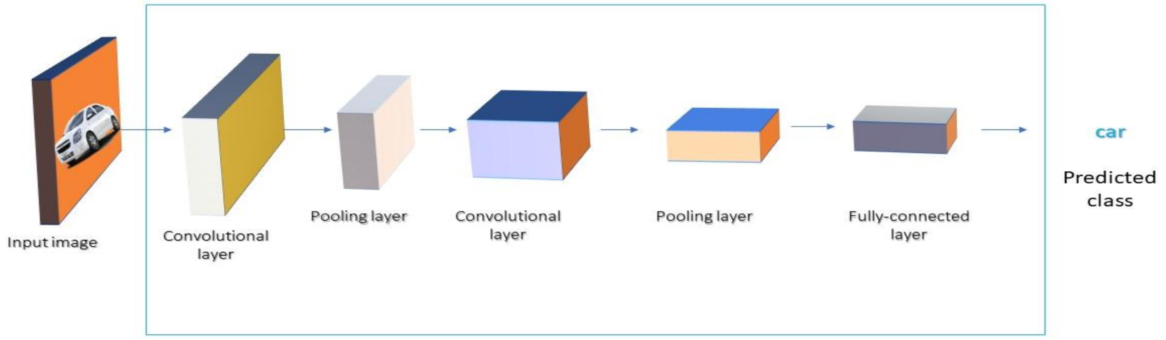


*Figure 1. ResNet-34 Model Architecture*

**Dataset**

This study used the Open Images Dataset V4 dataset, which contains 600 object classes and over 1,700,000 images. The dataset includes bounding boxes trained for object identification. This dataset is well suited for tasks such as image classification and object detection. In this study, the models were tested using data from the Open Images dataset for object classification and detection tasks. The dataset is split into training, validation, and testing sets with 70%, 10%, and 20% respectively. There is inequality in the number of objects between the classes, but this inequality is on a small scale. This dataset has been used in research and projects with a wide range of object classes in the Open Images dataset and successful results have been obtained.

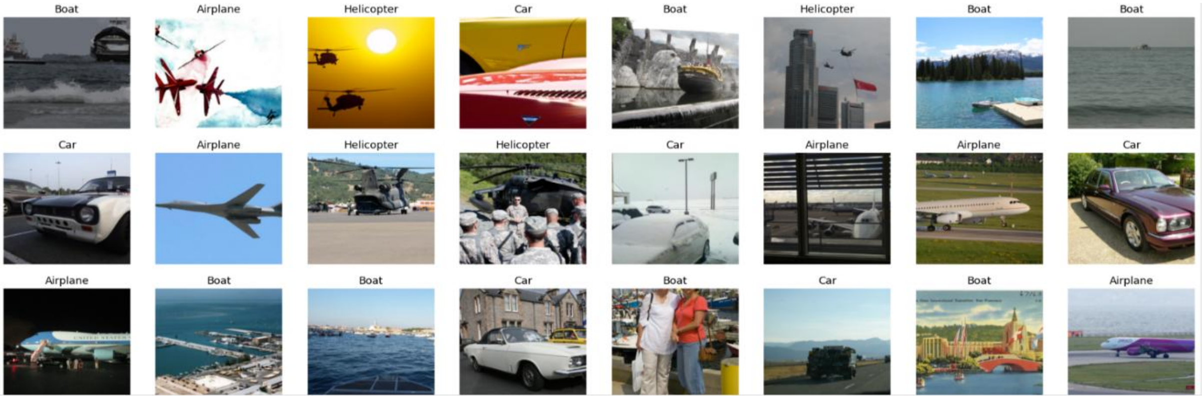


*Figure 2. Random pictures of the fourth class*

In this image, we can see the values of training loss, validation loss, accuracy, and elapsed time scores of the ResNet34 model across epochs in Table 1 below.

**Table 1.**

**Results of train loss, validation loss, accuracy and time in 20 epochs**

| epoch | train_loss | valid_loss | accuracy | time |
|---|---|---|---|---|
| 0 | 0.454523 | 0.442768 | 0.876068 | 01:05 |
| 1 | 0.316514 | 0.436259 | 0.879274 | 01:05 |
| 2 | 0.223195 | 0.457870 | 0.882479 | 01:06 |
| 3 | 0.190819 | 0.558902 | 0.872863 | 01:05 |
| 4 | 0.155241 | 0.518698 | 0.883547 | 01:07 |
| 5 | 0.140010 | 0.596468 | 0.870726 | 01:06 |
| 6 | 0.123535 | 0.536882 | 0.889957 | 01:04 |
| 7 | 0.100728 | 0.609098 | 0.865385 | 01:05 |
| 8 | 0.085246 | 0.595464 | 0.877137 | 01:05 |
| 9 | 0.079614 | 0.713256 | 0.870726 | 01:06 |

 

| | | | | |
|---|---|---|---|---|
| 10 | 0.064025 | 0.656306 | 0.875000 | 01:06 |
| 11 | 0.053691 | 0.566385 | 0.881410 | 01:05 |
| 12 | 0.039965 | 0.574271 | 0.886752 | 01:07 |
| 13 | 0.029993 | 0.553749 | 0.891026 | 01:05 |
| 14 | 0.027157 | 0.567435 | 0.888889 | 01:06 |
| 15 | 0.022961 | 0.551343 | 0.887821 | 01:05 |
| 16 | 0.018886 | 0.548102 | 0.894231 | 01:04 |
| 17 | 0.018463 | 0.546568 | 0.893162 | 01:07 |
| 18 | 0.016535 | 0.547200 | 0.897436 | 01:05 |
| 19 | 0.015260 | 0.559403 | 0.893162 | 01:05 |

From the table above, we can see that the class classification task during model training was performed using ResNet34. In the initial epoch, the model's training loss was 0.454, validation loss was 0.443, and the accuracy was 87.6%. In the subsequent epochs, the model showed significant improvements, with the training loss decreasing to 0.015 and the accuracy reaching 89.7% in the last epochs. Although the model's validation loss had some fluctuations, it was generally stable, which helped control the risk of overfitting, and the significant increase in accuracy and reduction in training loss in the 20th epoch indicated that the model was able to improve its learning performance during the training process.

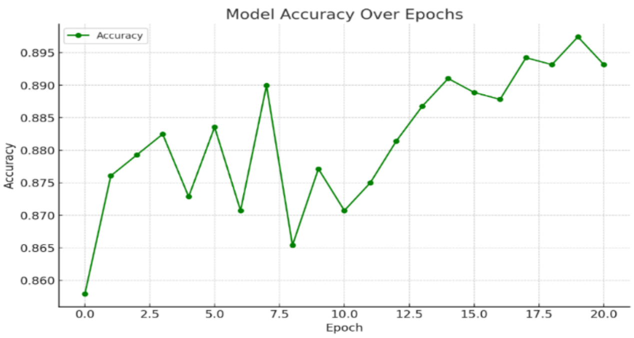


*Figure 3. Model accuracy across epochs.*

This plot shows how the accuracy of the model changed over each epoch. The accuracy is initially lower, with a significant increase in later epochs, but there is also a decrease in some time intervals. These changes show how the model adapts to the data during training, indicating that overfitting issues can sometimes occur. The plot shows that the accuracy reaches 0.897436 in the last epochs, meaning that the model fits the dataset well and the results are generally positive.

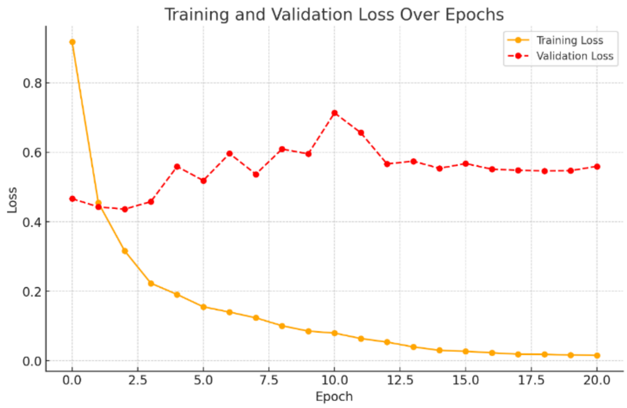


*Figure 4. Showing the training loss and validation loss*

The graph in Figure 4 shows the training loss and validation loss values during each epoch. As you can see in the graph, the training loss (yellow line) decreases quickly, which means that the model is adapting and learning on the dataset. And the validation loss (red line) remains relatively stable and high, which means that the model may have some problems generalizing to the data. This finding may indicate the possibility of overfitting, since the training loss decreases and the validation loss remains the same or increases. As this graph shows, the model outperforms the data and underperforms on the validation data.

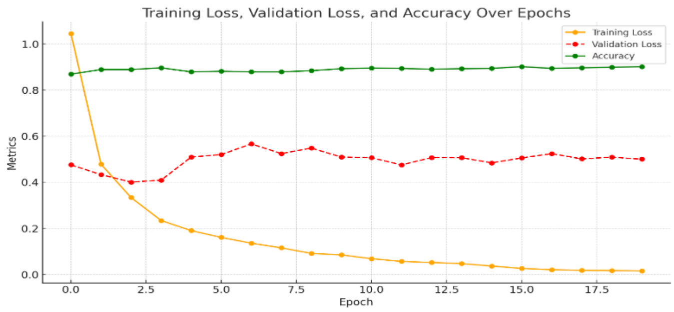


*Figure 5. Training Loss, Validation Loss, and Accuracy over Epochs*

 

 

Based on the graph and table above, we can conclude that during the training of the model, the training loss (train loss) started from 0.45 in the first epoch and decreased to 0.015 in the last epoch, indicating that the model learned the data well. The validation loss was 0.44 in the first epoch, and there was some fluctuation in the subsequent epochs, indicating some signs of overfitting. The accuracy started from 87.6% in the first epoch and reached 89.7% by the last epoch.

**Confusion Matrices**

A confusion matrix is an important tool for analyzing the performance of a model, especially in classification problems. A confusion matrix is used to determine how well or incorrectly a model responds. A confusion matrix shows the following:

- Y-axis (Actual): Shows the original actual data classes analyzed by the model.
- X-axis (Predicted): Shows the classes predicted by the model.
- Diagonal values: Indicates cases that the model classifies correctly, i.e. when the true class and the predicted class are the same.
- Off-diagonal values: Indicates cases where the model misclassifies.

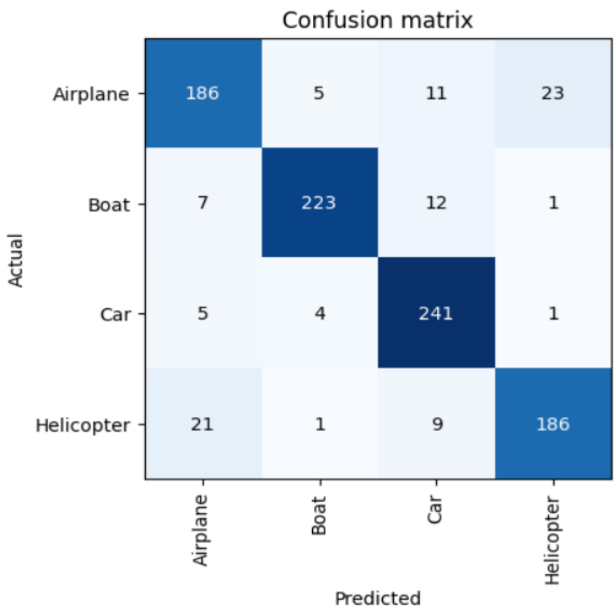


*Figure 6. Confusion Matrix*

Figure 6 above shows the correctly and incorrectly classified examples of the model classes. This matrix provides a visual representation of the model's performance evaluation and classification results.

**True Positive:** A True Positive (TP) is when the model correctly classifies a positive instance as positive.

**True Negative:** A True Negative (TN) is when the model correctly classifies a negative instance as negative.

**False Positive:** A False Positive (FP) is when the model incorrectly classifies a negative instance as positive.

**False Negative:** A False Negative (FN) is when the model incorrectly classifies a positive example as negative.

**Sensitivity (reminder):** Sensitivity is the proportion of True Positives among all True Positives, calculated using formula 1.

$$Recall = TP\,/\,(TP + FN) \qquad (1)$$

Accuracy: Accuracy is the percentage of correct positives among all predicted positives, calculated using formula 2.

$$Precision = TP\,/\,(TP + FP) \qquad (2)$$

F1 Score: A model is mainly evaluated by its F1 score because it provides more information than simple accuracy or average accuracy.

$$F1\ Score = (2 * P * R)/(P + R) \qquad (3)$$

Using these measurements, we can analyze which classes the model performs well in and which classes have more frequent misclassifications. Based on this information, we can use specific approaches to improve the model. Table 2 below shows the calculated values of the above measurements.

**Table 2.**

**F1 Score, Accuracy, and Class Recall Values in the Dimensions**

| **Model** | **Metric** | **Epoch Number** | **Classes** | | | |
|---|---|---|---|---|---|---|
| | | | Airplane | Helicopter | Boat | Car |
| ResNet-34 | **F1-score** | 20 | 0.838 | 0.869 | 0.937 | 0.920 |
| | **Precision** | 20 | 0.849 | 0.882 | 0.957 | 0.883 |
| | **Recall** | 20 | 0.857 | 0.918 | 0.960 | 0.857 |

When we look at the results, it is clear that the proposed model shows satisfactory results in all classes. When we compare the results between object classes, they are mostly close to each other. But the Boat class showed the best results in terms of F1-score (0.937), Recall (0.960), and Precision (0.957). Also in second place is the Car class, and the least successful classes are Airplane and Helicopter.

## CONCLUSION

The results of the study showed that the model developed using the ResNet-34 architecture can classify objects belonging to different classes, such as land, air and water vehicles, with high accuracy. This approach is more stable and efficient than traditional computer vision algorithms, and has shown excellent results in detecting vehicles even in complex environments.

This study has great practical significance and can be used in many fields, such as automated traffic management, security systems and drone monitoring. In particular, it is expected to make an important contribution to improving safety, reducing traffic congestion and accelerating the development of unmanned vehicles through automatic identification and classification of vehicles.

## References

1. X. Wang, Y. Zhao, and F. Pourpanah, "Recent advances in deep learning," Int. J. Mach. Learn. & Cyber., vol. 11, pp. 747–750, 2020. doi: 10.1007/s13042-020-01096-5
2. L. Aziz, M. S. B. Haji Salam, U. U. Sheikh, and S. Ayub, "Exploring Deep Learning-Based Architecture, Strategies, Applications and Current Trends in Generic Object Detection: A Comprehensive Review," in IEEE Access, vol. 8, pp. 170461-170495, 2020, doi: 10.1109/ACCESS.2020.3021508
3. S. Dargan, M. Kumar, M. R. Ayyagari, et al., "A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning," Arch Computat Methods Eng, vol. 27, pp. 1071–1092, 2020, doi: 10.1007/s11831-019-09344-w
4. H. Liu and B. Lang, "Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey," Applied Sciences, vol. 9, no. 20, p. 4396, 2019, doi: 10.3390/app9204396
5. S. Maity, A. Bhattacharyya, P. K. Singh, et al., "Last Decade in Vehicle Detection and Classification: A Comprehensive Survey," Arch Computat Methods Eng, vol. 29, pp. 5259–5296, 2022, doi: 10.1007/s11831-022-09764-1
6. M. Maity, S. Banerjee and S. Sinha Chaudhuri, "Faster R-CNN and YOLO based Vehicle detection: A Survey," 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2021, 10.1109/ICCMC51019.2021.9418274
7. S.Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology, Antalya, Turkey, 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186
8. Li Y. et al. Evaluation of decision fusions for classifying karst wetland vegetation using one-class and multi-class CNN models with high-resolution UAV images //Remote Sensing. – 2022. – T. 14. – №. 22. – C. 5869.
9. Avazov K. et al. Fire detection and notification method in ship areas using deep learning and computer vision approaches //Sensors. – 2023. – T. 23. – №. 16. – C. 7078.
10. Z. Shao, L. Wang, Z. Wang, W. Du and W. Wu, "Saliency-Aware Convolution Neural Network for Ship Detection in Surveillance Video," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, no. 3, pp. 781-794, 2020, doi: 10.1109/TCSVT.2019.2897980
11. S. Arabi, A. Haghighat, and A. Sharma, "A deep-learning-based computer vision solution for construction vehicle detection," Computer-Aided Civil and Infrastructure Engineering, vol. 35, Issue 7 , pp. 753-767, 2020, doi: 10.1111/mice.12530
12. J. Sang , Z. Wu , P. Gu , H. Hu, H. Xiang, Q. Zhangand B. Cai "An Improved YOLOv2 for Vehicle Detection", Sensors, vol. 18(12) ArticleID: 4272, 2018, doi: 10.3390/s18124272

13. Mittal S., Sangwan O. P. Classifying weather images using deep neural networks for large scale datasets //International Journal of Advanced Computer Science and Applications. – 2023. – Т. 14. – №. 1.
14. Inamullah et al. Data diversity in convolutional neural network based ensemble model for diabetic retinopathy //Biomimetics. – 2023. – Т. 8. – №. 2. – С. 187.
15. Pashaei M. et al. Review and evaluation of deep learning architectures for efficient land cover mapping with UAS hyper-spatial imagery: A case study over a wetland //Remote Sensing. – 2020. – Т. 12. – №. 6. – С. 959.
16. Chen L. et al. Review of image classification algorithms based on convolutional neural networks //Remote Sensing. – 2021. – Т. 13. – №. 22. – С. 4712.
17. Rawat W., Wang Z. Deep convolutional neural networks for image classification: A comprehensive review //Neural computation. – 2017. – Т. 29. – №. 9. – С. 2352-2449.
18. Subaar C. et al. Investigating the detection of breast cancer with deep transfer learning using ResNet18 and ResNet34 //Biomedical Physics & Engineering Express. – 2024. – Т. 10. – №. 3. – С. 035029.