Implementation and Evaluation of a Simple Convolutional Neural Network for Object Classification in Visual Assistance Systems

Raymond Pu *

Shanghai American School, Shanghai, China

* Corresponding Author Email: raymond01pd2026@saschina.org

Abstract. Image object recognition and classification has become more widespread in use and capable in functionality due to advancement in machine learning. One application of it is visual assistance systems for visually impaired persons, and there has been many existing proposed implementations and solutions about such systems. This article implements a simple convolutional neural network (CNN) machine learning model based on a simpler version of the architectures found in existing object detection models, which is trained using select data categories of the ImageNet dataset. The model is evaluated using sparse categorical accuracies, measuring the proportion of correct classifications, and comparing the number of predicted and expected classifications for each data category. The accuracy of the model did not perform well as expected because of significant overfitting behavior noticed through the validation loss. Even if early stopping to reduce overfitting is implemented, the overall accuracies still cannot be fully remediated. However, the difference between the number of expected and actual predictions is comparable.

Keywords: Machine learning; Image classification; Visual assistance systems; Convolutional neural network.

1. Introduction

Over the course of technological development, object recognition and classification in images has become more versatile and capable, and have been used in a wide variety of applications and implementations from image labeling in photo software to autonomous vehicle driving. It is made possible by the advancement in machine learning and deep learning technologies. Because of this advancement, it is now possible to apply object recognition to real time camera and video data due to a machine learning architecture named Convolutional Neural Network (CNN) [1], opening it to a further range of applications and possibilities.

Visual assistance systems for visually impaired persons are one of the applications of this technology. There have been many proposed implementations and solutions utilizing object detection technology [2,3,4]. These systems help these people with disabilities identify objects and obstacles in their surroundings, improving their quality of life. Some of them propose hardware implementations of such systems, while others propose improved architectures to the machine learning models used for object detection. Besides object detection, these systems perform a variety of other assistive tasks to accommodate for disabilities, such as detecting uneven road surfaces, ramps, and stairways.

The purpose of this paper is to create and evaluate the accuracy a simple CNN image classification model based on network architectures of existing implementations using images of common obstacles and objects related or beneficial to visual assistance systems. This paper will first cover existing implementations and past works referenced during design and research, present the method and implementation, and evaluate the accuracy of the implementation.

2. Related works

This research task references many past works and implementations of image recognition and obstacle detection machine learning models. The design of the machine learning model used in this research task is derived from these implementations.

First, the architecture of several image object detection models is referenced. These include the Darknet neural network in YOLO9000 and YOLOv3 [5,6]. These networks include several conventional, max pooling, and residual layers for feature extraction and image classification, consisting of multiple groups. Each group consists of 2 or more convolutional layers with alternating sizes, along with a residual or max pooling layer at the end. The object detection models also consist of other algorithms for location box clustering of the object and performance optimizations, supporting the detection of multiple objects in a single image and their respective positions. The architecture of the neural network used for the classification functionality is referenced for this task and a simpler and similar design is used for the implementation.

Articles proposing an implementation for obstacle detection systems are also referenced for this task. For example, Salavati, Pouyan and Hossein Mahvash Mohammadi uses the UnspVGG16 neural network architecture for global feature extraction, which is used for determining and classifying whether the image block contains an obstacle or object [3]. It is based on the VGG16 architecture with an unsupervised learning algorithm. This architecture consists of multiple connected groups of layers each consisting of 2-3 convolutional and a max pooling layer, and the output size increases over the groups. The architecture of this neural network is also referenced and considered during the research process of this task as a potential implementation of the model.

Sudharshan, Duth P., and Swathi Raj uses a simple CNN model for recognizing objects in images [7]. The network is made of two 2D convolutional layers, 1 max pooling layer, 1 dropout layer, and dense layers at the end, which is much simpler than the architecture of other neural networks in above articles. However, this article uses TensorFlow deep learning Python library in its experiment, which provides useful information for the implementation of this task.

Yadav, Saumya, et al also uses the YOLOv3 model for the object detection implementation of the proposed device [2]. The Darknet architecture of the model is also detailed in this article, which is referenced by this task. This article also details a hardware and device implementation and its sensor arrangement, adapting the algorithm for real world use and making it suitable for a wide range of obstacle detection scenarios in the real world.

Radovic, Matija, Offei Adarkwa, and Qiaosong Wang uses a CNN model and algorithm based on the architecture used by YOLO for the object detection implementation [1]. The primary objective of this article is to test and evaluate such algorithms for real-time object detection and classification in UAV operations. The model is evaluated using satellite images of airplanes and aircraft.

In conclusion, this task attempts to create a simple machine learning model similar to the architectures of other existing implementations. This task attempts to explore and evaluate such implementation through the experimentation process.

3. Approach and Implementation

First, to better represent the types of images and data mostly used for assistive applications, data with specific labels and classes are selected and filtered from the ImageNet image classification dataset [8,9]. The selected labels and classes include vehicles, street features (such as street signs, traffic lights, and poles), bicycles/bikes, and shops and convenience stores. These labels are grouped into these four categories for evaluation and processing. This dataset filtering is applied to both the training and testing data used during the model training process. Figure 1 shows some of the images from the selected and filtered training data.

The filtered data is then used to train a CNN based on existing underlying designs and architectures of other image classification models, such as Darknet in YOLO [3,5,7], as shown in Figure 3. Even though the design of the CNN model is similar to these architectures, the amount and complexity of

the layers are simpler due to the smaller size of the dataset and the input data (64x64 instead of 256x256), as shown in Table 1. There are 42,816 training samples in total in the filtered training data, consisting of 33 labels and classes in total.

After the model is trained using the training data, the models are evaluated by comparing the groups of the expected and predicted labels as detailed above by selecting 5000 random samples from the entire dataset. In this evaluation process, the testing sample is considered a match when the corresponding group of the expected and predicted label of the sample are the same. The selected labels and classes are organized into the following groups as shown in Figure 2.

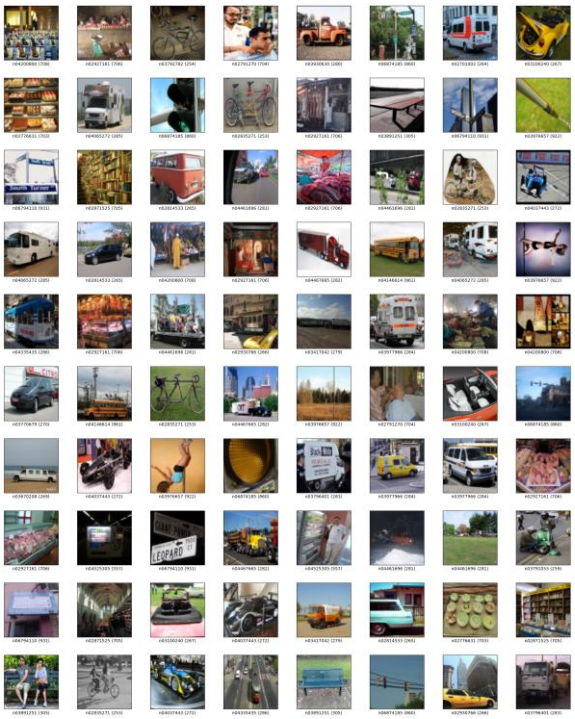

Fig. 1 (left) Selected training samples from the ImageNet dataset

```
filter_items={
    # Vehicles
    "n03393912": "freight_car",
    "n02701002": "ambulance",
    "n02814533": "beach_wagon",
    "n02930766": "cab",
    "n03100240": "convertible",
    "n03594945": "jeep",
    "n03670208": "limousine",
    "n03770679": "minivan",
    "n03777568": "Model_T",
    "n04037443": "racer",
    "n03417042": "garbage_truck",
    "n03930630": "pickup",
    "n04461696": "tow_truck",
    "n04467665": "trailer_truck",
    "n03796401": "moving_van",
    "n03977966": "police_van",
    "n04065272": "recreational_vehicle",
    "n04335435": "streetcar",
    "n04146614": "school_bus",
    # Street features
    "n06794110": "street_sign",
    "n06874185": "traffic_light",
    "n03976657": "pole",
    "n03891251": "park_bench",
    "n04525305": "vending_machine",
    # Street-parked bicycles and bikes
    "n02835271": "bicycle-built-for-two",
    "n03792782": "mountain_bike",
    "n03791053": "motor_scooter",
    # Shops and convenience stores
    "n03461385": "grocery_store",
    "n02776631": "bakery",
    "n02791270": "barbershop",
    "n02871525": "bookshop",
    "n02927161": "butcher_shop",
    "n04200800": "shoe_shop",
}
```

Fig. 2 (right) Group organization of dataset labels

Table 1. CNN model used for image classification

Layer	Filters	Shape
Convolutional	32	64x64
Convolutional	64	64x64
Max pooling	64	32x32
Dropout (20%)	64	32x32
Convolutional	128	32x32
Dropout (20%)	128	32x32
Convolutional	64	32x32
Convolutional	128	32x32
Dropout (20%)	128	32x32
Convolutional	64	32x32
Convolutional	128	32x32
Max pooling	128	16x16
Dropout (20%)	128	16x16
Flatten	32768	
Dense	512	
Dropout (20%)	512	
Dense	1000	

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Fig. 3 Darknet-19 network used in YOLO9000

After the model is trained using the training data, the models are evaluated by comparing the groups of the expected and predicted labels as detailed above by selecting 5000 random samples from

the entire dataset. In this evaluation process, the testing sample is considered a match when the corresponding group of the expected and predicted label of the sample are the same. The selected labels and classes are organized into the following groups as shown in the table below.

4. Experiment results

After training the model, the accuracy and loss during the training and validation process are compared. During initial training, 15 epochs was used to train the model. However, this setting resulted in significant overfitting, as shown in Figure 4. To resolve this issue, the early stopping technique was used when re-training the model, reducing visible overfitting during the training process and resulted in only 7 epochs being used. However, the training accuracy is negatively affected by this technique due to the shortage of epochs, with only 65% compared to 90% as shown in Figure 5. The accuracy and loss shown in these figures are calculated using the sparse categorical accuracy and cross entropy algorithm [10]:

$$J(\boldsymbol{w}) = -\frac{1}{N}\sum_{i=1}^{N}[y_i log(\hat{y_i}) + (1-y_i)log(1-\hat{y_i})] \quad (1)$$

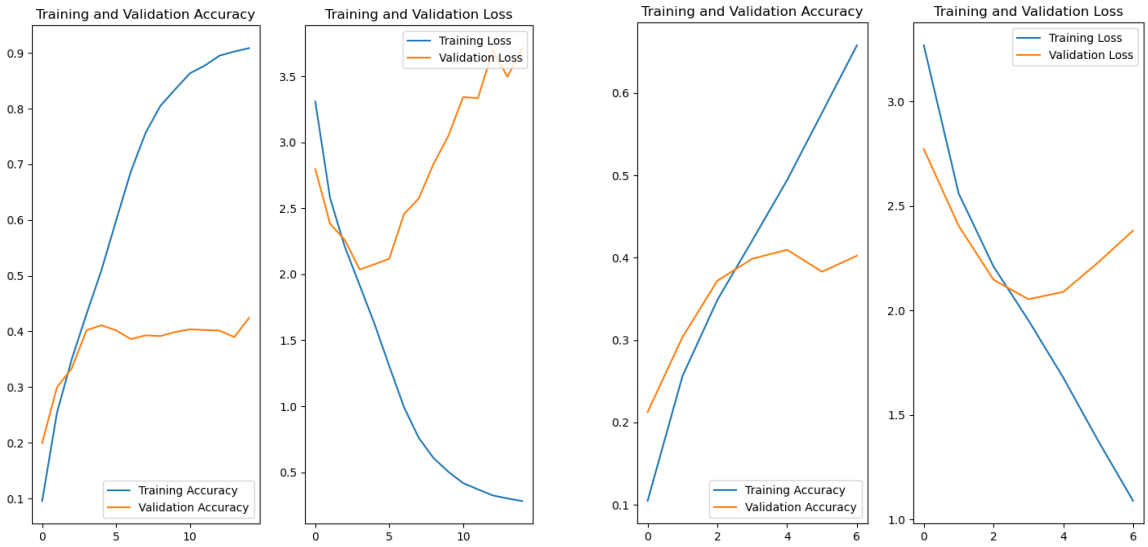

Fig. 4 (left) Training process with overfitting

Fig. 5 (right) Training process with early stopping implemented

Then, the model is evaluated using the method detailed previously. It can be observed in Table 2 and Table 3 that the difference between the number of expected samples and the number of predicted samples classified into the four groups are relatively small compared to the total amount of samples. However, the category accuracy, the percentage of testing samples categorized into the same category as the expected label, is much lower than expected. In fact, this accuracy is very similar to the validation accuracy shown during the training process. It is also noticed that the difference and the category accuracy evaluated using the models with and without overfitting, which is also reflected in the graph visualizations of the validation accuracy of both models.

Table 2. Result from model with early stopping

	Expected	Predicted	Difference
Vehicles	2896	2922	26
Street features	752	737	15
Bicycles and bikes	432	443	11
Shops and convenience stores	920	898	22

Table 3. Result from model with overfitting

	Expected	Predicted	Difference
Vehicles	2897	2902	5
Street features	762	706	56
Bicycles and bikes	470	434	36
Shops and convenience stores	871	958	36
Category accuracy	**39.08%**		

5. Conclusion

Due to the accuracy and performance issues the model experienced during training and evaluation, the model did not perform as well as expected, compared to results of other past works. Several reasons might cause such issues, such as the design of the model is too complex for the input size, lack of training data due to category filtering, or the suitability of the chosen architecture. The results have shown that this approach might not work the best for this image dataset or resolution, and other optimizations and improvements are needed on top of this implementation. Alternatively, other model architectures, such as VGG16, can be used to reduce training inaccuracies such as overfitting, which is an improvement noticed over the Darknet architecture but was not chosen due to lower yield of training accuracy.

References

[1] Radovic M, Adarkwa O, Wang Q. Object recognition in aerial images using convolutional neural networks. Journal of Imaging, 2017, 3(2): 21.

[2] Yadav S, Joshi R C, Dutta M K, et al. Fusion of object recognition and obstacle detection approach for assisting visually challenged person//2020 43rd International Conference on Telecommunications and Signal Processing (TSP). IEEE, 2020: 537-540.

[3] Salavati P, Mohammadi H M. Obstacle detection using GoogleNet//2018 8th international conference on computer and knowledge engineering (ICCKE). IEEE, 2018: 326-332.

[4] Trabelsi R, Jabri I, Melgani F, et al. Indoor object recognition in RGBD images with complex-valued neural networks for visually-impaired people. Neurocomputing, 2019, 330: 94-103.

[5] Redmon J, Farhadi A. YOLO9000: better, faster, stronger//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.

[6] Redmon J. Yolov3: An incremental improvement. arxiv preprint arxiv:1804.02767, 2018.

[7] Sudharshan D P, Raj S. Object recognition in images using convolutional neural network//2018 2nd International Conference on Inventive Systems and Control (ICISC). IEEE, 2018: 718-722.

[8] Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database//2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009: 248-255.

[9] Chrabaszcz P, Loshchilov I, Hutter F. A downsampled variant of imagenet as an alternative to the cifar datasets. arxiv preprint arxiv:1707.08819, 2017.

[10] "Cross-Entropy Cost Functions Used in Classification." GeeksforGeeks, 11 Oct. 2021, www.geeksforgeeks.org/cross-entropy-cost-functions-used-in-classification/.