CSC345 – Coursework: Object Recognition 2020537

1. Introduction

Object recognition is an essential task in many applications, including self-driving automobiles, surveillance systems, and image recognition systems. Using the CIFAR100 dataset, which contains images of 100 different types of objects, we provide our evaluation research on the problem of object recognition in this report. We intend to use the training set to create a classifier to identify the objects in the testing set precisely. The dataset has been split into training and testing sets.

We used three different machine learning algorithms to categorize the objects in the images in order to complete this task: a deep convolutional neural network (CNN), a CNN based on visualized Histogram of Oriented Gradients (HOG) features, and a deep neural network based on HOG features extracted from the images. In order to establish which algorithm performs the task the best, we tested its performance on the CIFAR100 dataset's fine and coarse labels.

In the following sections of this report, we will go into more depth about each machine learning algorithm, show the findings of our experimental research, and offer a critical assessment of the approaches' advantages and disadvantages. We will also discuss potential directions for future research using the CIFAR100 dataset to enhance object recognition.

2. Methodology

CNN

Convolutional neural networks (CNNs) have been widely used for image classification tasks due to their ability to learn complex data patterns through convolutional layers. We chose to use a deep CNN as our first method, as deeper CNNs have been shown to achieve better performance on image classification tasks due to their increased capacity to model the data [1].

Our CNN architecture consisted of a series of convolutional layers, max-pooling layers, and fully-connected (dense) layers, with rectified linear unit (ReLU) activation functions [2] used in the convolutional and dense layers. We included dropout layers in the model to prevent overfitting [3], and we experimented with different levels of dropout in order to achieve a balance between overfitting and general accuracy.

The CNN had a total of four convolutional layers, with kernel sizes of 3x3 and 32, 64, and 128 filters. The max pooling layers had a pool size of 2x2. We also included dropout layers after each convolutional and max pooling layer, with dropout rates ranging from 0.1 to 0.25. We used batch normalization layers [4], [5] after each convolutional layer to normalize the activations and improve the training process.

For the fine-label classification task, the fully-connected layers of the CNN were composed of a flatten layer followed by a dense layer with 256 units and a dropout rate of 0.5. The output layer had 100 units, one for each of the fine label classes in the CIFAR100 dataset. For the coarse label classification task, the output layer had ten units, one for each coarse label class in the dataset.

We used the Adam optimizer with default hyperparameters for training the model. The Adam optimizer [6] is a gradient-based optimization algorithm that adapts the learning rates of the model's parameters based on the historical gradient information. It has been shown to be effective for a wide range of tasks and is often used as a default choice for training deep learning models.

For the loss function, we used categorical cross-entropy [7], which is a common choice for multi-class classification tasks. Categorical cross-entropy is a measure of the difference between the predicted class probabilities and the true class probabilities, and it is minimized during training to achieve good performance on the classification task.

Overall, the first method consisted of a deep CNN with a complex architecture and several regularization techniques to prevent overfitting. The model was trained for 70 epochs with a batch size of 50, and the results were evaluated on the testing set for both the fine and coarse label classification tasks.

CNN + HOG

A CNN based on visually represented Histogram of Oriented Gradients [8] (HOG) features was the second technique we employed. HOG features are a class of feature descriptors that depict the distribution of gradient orientations in an image to describe the shape and texture of objects in the image. We chose to use visualized HOG features as input to the CNN in order to test whether using pre-processed features as input to the CNN would improve the object recognition performance compared to using raw image data.

To extract the HOG features, we used skimage.feature.hog [8] function on the images, resulting in a histogram of the orientations of the gradients in the image. We then visualized the HOG features by plotting the histogram values as an image, which can give some insights into the shape and texture of the objects in the image. We used the HOG visualization as input to the CNN, rather than the raw HOG features.

The CNN architecture for this method was almost the same the first method, with several convolutional and max pooling layers, as well as dropout and batch normalization layers to prevent overfitting. However, the input shape of the CNN was changed to (32,32,1) to match the shape of the HOG visualizations.

We used the Adam optimizer with default hyperparameters for training the model, and the categorical cross-entropy loss function as the objective to be minimized. The model was trained for 70 epochs with a batch size of 50, and the results were evaluated on the testing set for both the fine and coarse label classification tasks.

For the HOG feature extraction, we used the default values of "pixels_per_cell=[5,5], cells_per_block=[5,5]" for the scikit-image library's HOG function. These features were relatively fast to compute with those parameters, but it is possible that using different values may have resulted in improved performance, at cost of longer computation time. Overall, the second method involved using HOG visualizations as input to a CNN for object recognition, and the results were compared to those of the first method to assess the potential benefits of using pre-processed HOG features.

DNN + HOG

A deep neural network (DNN) based on a Histogram of Oriented Gradients (HOG) features was our third method for object recognition. This technique required feeding a DNN for object recognition with HOG characteristics that were retrieved from the photos. The HOG features record data on the gradient orientations in the photos, which may be used to categorize things according to their texture and shape.

To extract the HOG features, we used skimage.feature.hog [8] function on the images, resulting in a histogram of the orientations of the gradients in the image. The HOG features were already flattened and had a shape of (900,) when they were returned from the feature extraction function.

For the coarse label classification task, the DNN architecture consisted of three fully connected layers with 256, 128, and 64 nodes, respectively. Each layer had a ReLU activation function and was followed by a batch normalization layer. We also included dropout layers with a rate of 0.2, 0.3, and 0.4, respectively, to attempt to prevent overfitting. However, despite these regularization techniques, the DNN still showed signs of overfitting after only 10 epochs of training. The input to the DNN was the HOG features, and the output was a prediction of the object class.

For the fine label classification task, the DNN architecture was similar, but the number of nodes in the final layer was increased to 100 to match the number of fine labels. Two layers have 512 and 256 nodes, respectively. The same regularization techniques were used, but the model still overfitted the data after 10 epochs of training.

We used the Adam optimizer with default hyperparameters for training the model, and the categorical cross-entropy loss function as the objective to be minimized. The model was trained for 40 epochs with a batch size of 50, and the results were evaluated on the testing set for both the fine and coarse label classification tasks.

The third technique generally included feeding a DNN with HOG features to recognition objects. The DNN showed symptoms of overfitting the data despite the use of regularization approaches, which may have led to the lower performance relative to the other methods.

3. Results

The performance of three distinct algorithms on the CIFAR100 dataset was assessed in this section. The first technique classified the objects using a deep convolutional neural network (CNN). The third technique employed a deep neural network based on HOG features, while the second method used a CNN based on visualized Histogram of Oriented Gradients (HOG) features. The table below provides a summary of the three approaches' findings:

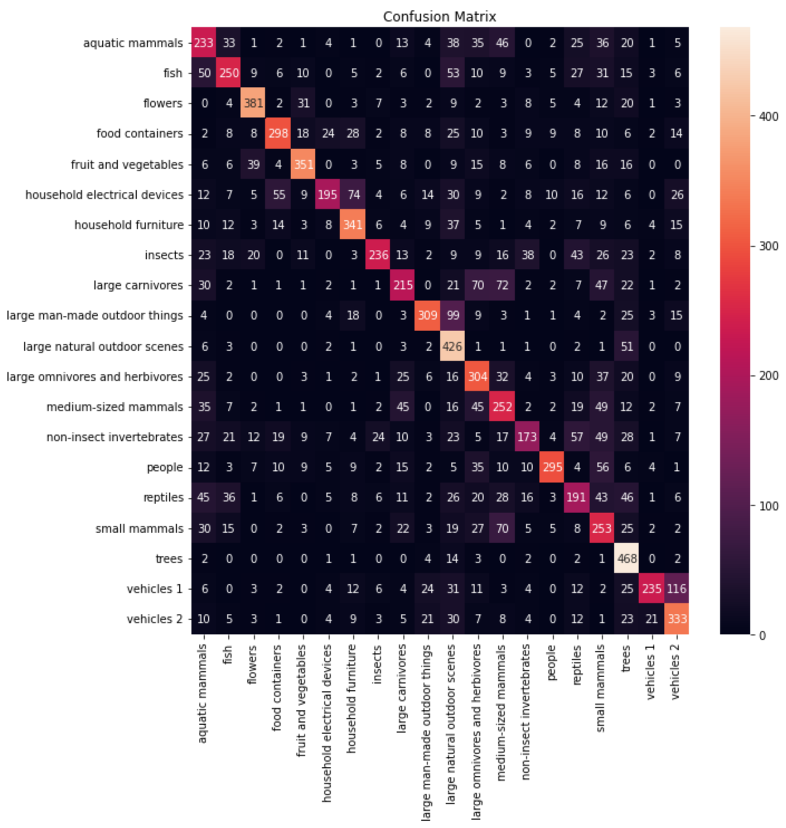
Method	Test Accuracy (Fine Labels)	Loss (Fine Labels)	Test Accuracy (Coarse Labels)	Loss (Coarse Labels)
1 (CNN)	0.4625	2.0575	0.5739	1.3764
2 (CNN + HOG)	0.2615	3.1258	0.3816	2.0265
3 (HOG + DNN)	0.2377	3.3577	0.3437	2.1634

The first technique, which employed a pure CNN, displayed the highest test accuracies and the lowest losses among the three methods on the fine and coarse label classification tasks. The second technique, which used a CNN based on HOG features, performed worse on both tasks in terms of test accuracies and losses. The third technique, which made use of a deep neural network based on HOG characteristics, had the worst test accuracies as well as the most losses.

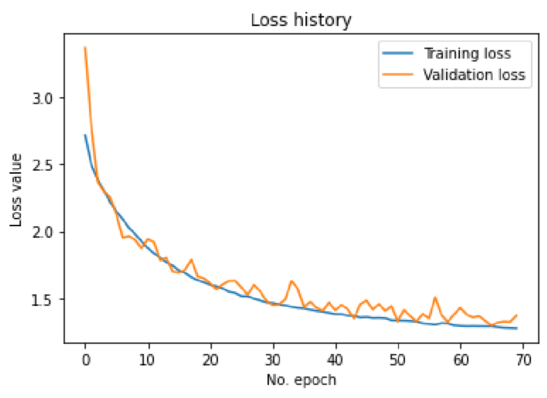
The second and third algorithms may have performed poorly partly because HOG features may not be as useful for object categorization as pure CNNs [1]. Alternately, it is likely that these methods' neural networks were not best suited to benefit from HOG features. In any event, it is evident that the first strategy, which made use of a single CNN, was the most successful one when it came to classifying objects on the CIFAR100 dataset.

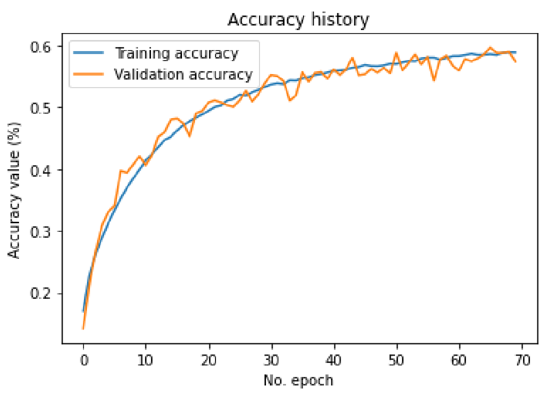
The confusion matrix for the first method on the coarse label classification task is shown below.

In addition to the confusion matrix, we also included plots of the loss and accuracy for the first method on both the fine and coarse label classification tasks. These plots show that the model was able to achieve a relatively low loss and high accuracy on both tasks, indicating that it was able to classify the objects in the CIFAR100 dataset effectively. Overall, the first method performed the best out of the three methods, with the highest test accuracies and the lowest losses on both tasks. While the second and third methods also used HOG features, they did not perform as well as the first method, possibly due to the design of the neural networks or the fact that CNNs are generally more effective at object recognition tasks [1].



	aquatic mammals	fish	flowers	food containers	fruit and vegetables	household electrical devices	household furniture	insects	large carnivores	large man-made outdoor things	large natural outdoor scenes	large omnivores and herbivores	medium-sized mammals	non-insect invertebrates	people	reptiles	small mammals	trees	vehicles 1	vehicles 2
aquatic mammals	233	33	1	2	1	4	1	0	13	4	38	35	46	0	2	25	36	20	1	5
fish	50	250	9	6	10	0	5	2	6	0	53	10	9	3	5	27	31	15	3	6
flowers	0	4	381	2	31	0	3	7	3	2	9	2	3	8	5	4	12	20	1	3
food containers	2	8	8	298	18	24	28	2	8	8	25	10	3	9	9	8	10	6	2	14
fruit and vegetables	6	6	39	4	351	0	3	5	8	0	9	15	8	6	0	8	16	16	0	0
household electrical devices	12	7	5	55	9	195	74	4	6	14	30	9	2	8	10	16	12	6	0	26
household furniture	10	12	3	14	3	8	341	6	4	9	37	5	1	4	2	7	9	6	4	15
insects	23	18	20	0	11	0	3	236	13	2	9	9	16	38	0	43	26	23	2	8
large carnivores	30	2	1	1	1	2	1	1	215	0	21	70	72	2	2	7	47	22	1	2
large man-made outdoor things	4	0	0	0	0	4	18	0	3	309	99	9	3	1	1	4	2	25	3	15
large natural outdoor scenes	6	3	0	0	0	2	1	0	3	2	426	1	1	1	0	2	1	51	0	0
large omnivores and herbivores	25	2	0	0	3	1	2	1	25	6	16	304	32	4	3	10	37	20	0	9
medium-sized mammals	35	7	2	1	1	0	1	2	45	0	16	45	252	2	2	19	49	12	2	7
non-insect invertebrates	27	21	12	19	9	7	4	24	10	3	23	5	17	173	4	57	49	28	1	7
people	12	3	7	10	9	5	9	2	15	2	5	35	10	10	295	4	56	6	4	1
reptiles	45	36	1	6	0	5	8	6	11	2	26	20	28	16	3	191	43	46	1	6
small mammals	30	15	0	2	3	0	7	2	22	3	19	27	70	5	5	8	253	25	2	2
trees	2	0	0	0	0	1	1	0	0	4	14	3	0	2	0	2	1	468	0	2
vehicles 1	6	0	3	2	0	4	12	6	4	24	31	11	3	4	0	12	2	25	235	116
vehicles 2	10	5	3	1	0	4	9	3	5	21	30	7	8	4	0	12	1	23	21	333





4. Conclusion

In conclusion, we evaluated the performance of three different methods for object recognition on the CIFAR100 dataset. The first method, which used a deep convolutional neural network (CNN), performed the best, with higher test accuracies and lower losses on both the fine label and coarse label classification tasks. The second method, which used a CNN based on visualized Histogram of Oriented Gradients (HOG) features, had lower test accuracies and higher losses than the first method on both tasks. The third method, which used a deep neural network based on HOG features, had the lowest test accuracies and the highest losses of the three methods on both tasks.

As future work, it may be worth exploring other types of neural networks for image classification such as Vision transformer [9], as well as reconsidering the use of HOG features for this task. Additionally, further analysis and optimization of the design of the neural networks used in this study could potentially lead to improved performance.

5. Bibliography:

[1] G. K V and J. Gripsy, *Image Classification using HOG and LBP Feature Descriptors with SVM and CNN*. 2020.

[2] A. F. Agarap, 'Deep Learning using Rectified Linear Units (ReLU)'. arXiv, Feb. 07, 2019. Accessed: Dec. 19, 2022. [Online]. Available: http://arxiv.org/abs/1803.08375

[3] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 'Dropout: A Simple Way to Prevent Neural Networks from Overfitting'.

[4] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, 'Understanding Batch Normalization', in *Advances in Neural Information Processing Systems*, 2018, vol. 31. Accessed: Dec. 18, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/36072923bfc3cf47745d704feb489480-Abstract.html

[5] S. Ioffe and C. Szegedy, 'Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift'.

[6] K. K. Kumar and E. Al, 'An Efficient Image Classification of Malaria Parasite Using Convolutional Neural Network and ADAM Optimizer', *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 2, Art. no. 2, Apr. 2021, doi: 10.17762/turcomat.v12i2.2398.

[7] 'Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names'. https://gombru.github.io/2018/05/23/cross_entropy_loss/ (accessed Dec. 19, 2022).

[8] 'Histogram of Oriented Gradients — skimage v0.19.2 docs'. https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_hog.html (accessed Dec. 19, 2022).

[9] A. Dosovitskiy *et al.*, 'An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale'. arXiv, Jun. 03, 2021. Accessed: Oct. 29, 2022. [Online]. Available: http://arxiv.org/abs/2010.11929