Coursework: Object Recognition

Student Number: 2042274

Module: CS345

Swansea University
Prifysgol Abertawe

Department of Computer Science

26th December 2022

Table of Contents

1 Introduction

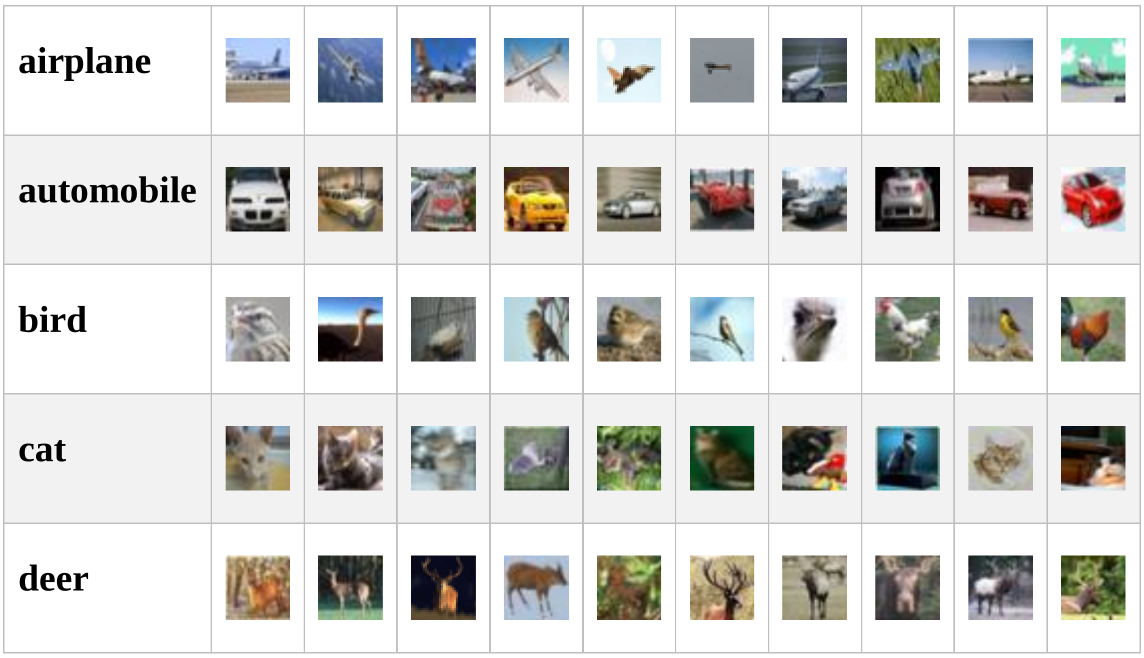
We have seen in the past few decades a rampant rise in digital image data being available. This is mainly because of easily accessible and affordable devices with low maintenance. Undoubtedly, this has led to heavy demand for image recognition and classification for better maintenance of such a large amount of image data. Image recognition also has other great applications like that in healthcare, security systems, e-commerce etc. Academic, as well as industrial research in image recognition, has been going on for quite a few years now, and it is still a challenging and open problem for researchers. In this paper, we will try to give a glimpse of the solution to this problem using a dataset and also discuss different methods and their shortcomings and ways to improve them.

The problem we will be looking at is quite simple in terms of understanding. The dataset used has 10 different categories of images with each category containing a number of sample images. Based on this, we need to come up with an algorithm to detect images and identify what category it belongs to.

We are using a subset of the CIFAR-10 dataset[1] for this problem. The proposed methodology used is based on Convolutional Neural Networks (CNN). It is achieved by training a model using the training images and labels and tweaking and adding on a few layers to achieve good accuracy.

2 Method:

The CIFAR-10 dataset provided to us consists of 32x32 colour images. There are 10,000 training images and 1000 test images. It has 10 different classes of images i.e., airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck.



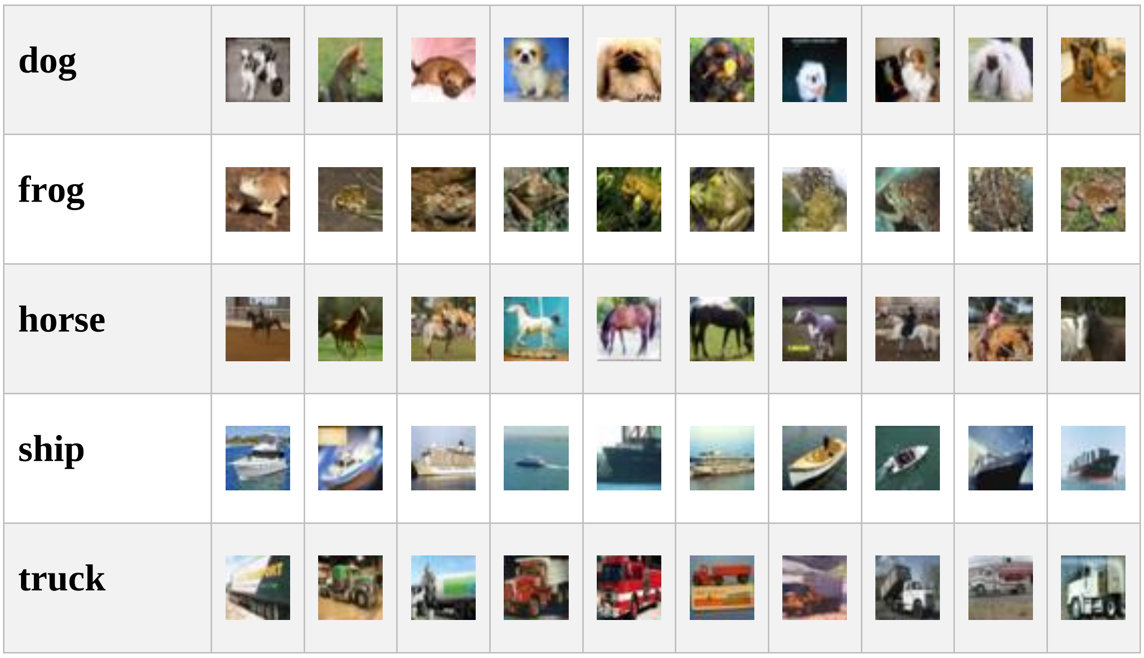


Fig. 1: Different classes in the dataset[1]

2.1 Convolutional neural network (CNN) vs other classification algorithms

For our problem, we will be using the convolutional neural network algorithm (CNN) to analyse the data and carry out the classification task. A (non-convolutional) fully-connected neural network is not a good approach for this task as it is sensitive to the location of an object and it also requires much more computational power compared to CNN[2]. CNNs are invariant to moderate translation of objects in images[3] and also require less computational power due to the process of "pooling layers" which reduces the dimension of data.

In CNN, we do not need to do any feature extraction from the images manually as the algorithm does that by itself using kernels or convolutional matrices in the convolutional neural layers. Through the process of automated learning, the CNNs can learn to optimize the kernels efficiently[4]. An artificial neural network without convolution would require feature extraction from the images for a fair accuracy goal. Other classifier algorithms such as support vector machines would also require feature extraction and processing of the dataset.

2.2 Building our CNN model

In order to build our convolutional neural network model, we would first create a Sequential model object. As our problem involves identifying objects in complex images, this model will consist of three convolutional layers and three fully connected neural layers (including the final output layer) for achieving good accuracy on the test dataset. The number of learnable filters or kernels in each of the convolutional layers would be 16, 32 and 64 for the first, second and third layers respectively. We are gradually increasing the number of filters in each layer because as we move forward in the layers, the feature maps get more complex and there are larger combinations of patterns to pick up. Each of the convolutional layers would have a kernel size of (3,3) and Rectified Linear activation applied. After each convolutional layer, a max pooling layer would be applied with pool size (2,2) and the default value of stride = 1. Pooling is a form of non-linear down-sampling of the input data which helps in reducing the computational power required for the model to train[2]. After the convolutional and pooling layers, the output layer is flattened and the final classification is done through another three fully-connected neural layers. The number of neurons in these layers is 128, 64, and 10 for the fourth, fifth and last being the output layer. The first two hidden fully- connected neural layers have Rectified Linear activation applied whilst the output layer has a “*softmax*” activation so as to place the values into the probability space[5].

We then compile our model to be trained. In the compile method, we use the “*adam*” optimizer, loss as “*sparse_categorical_crossentropy*” as the targets are labels and are not one-hot encoded and, the metrics as “*accuracy*” as again the targets are labels[5].

```
cnn = models.Sequential([
    #cnn layer1
    layers.Conv2D(filters=16, kernel_size=(3, 3), activation='relu', input_shape=(32, 32, 3)),
    layers.MaxPooling2D(pool_size=(2, 2), strides=1),

    #cnn layer2
    layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    #cnn layer3
    layers.Conv2D(filters=64, kernel_size=(3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    #ann layers
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation='softmax')
])
```

```
cnn.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics='accuracy')
```

Fig. 2 The CNN model

3 Results

After training the model on the training dataset and then evaluating the accuracy of the model on the test dataset, it comes out to be 53.1%. This score is very acceptable given the fact that we did not go through any feature extraction or processing of the training image dataset. This is in part due to the fact that we used CNN in our model which does that by itself. Other methodologies like support vector machines or fully-connected neural networks would have required the huge task of image feature extraction to achieve this range of accuracy.

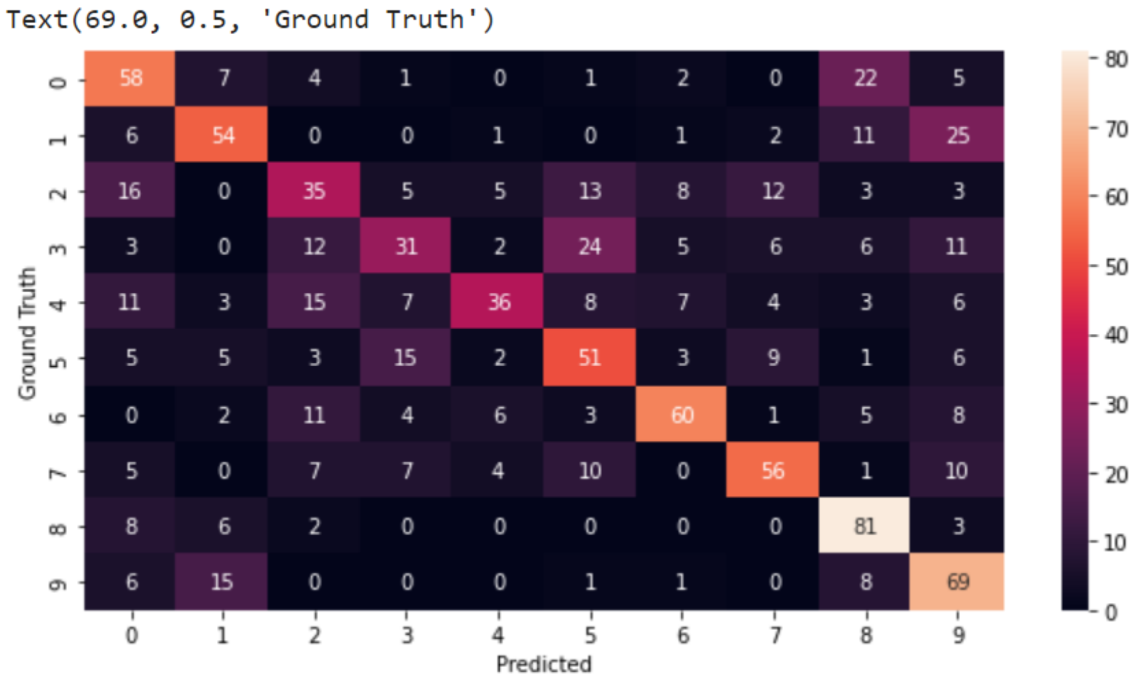


Fig. 3 Confusion matrix on the test dataset

In the above confusion matrix for the test dataset, we observe that the class with index 0 (i.e., *airplane*) was correctly predicted 58 times by the model, 54 correct predictions for class with index 1, 35 for index 2 and so on. We can also look at *automobile (index no. 1)* which was wrongly classified as *airplane* 6 times, *bird (index no. 2)* as *airplane* 16 times and so on.

The class with index 8 (i.e., *ship*) has the most correct classification by the model followed by the class with index 9 (i.e., *truck*) whereas *automobile (index no. 1)* was most wrongly classified as *truck (index no. 9)* 25 times.

4 Conclusions

In this paper, we attempted the task of classifying images based on the object it contains. We used a CNN for our model which helped us not to bother about feature extraction and processing of the image dataset. If we had implemented some other classifier algorithm, we have had to deal with the task of feature extraction from the raw data and then train our model using those extracted features. We achieved an accuracy of 53.1% on the testing data. Although it looks good, this range is still not acceptable for the majority of daily-task as well as on commercial levels. Our model would require further tweaking and image manipulation for achieving a better accuracy rate. There are several limitations to our model. CNNs cannot handle the orientation and scaling of objects by themselves. This can be handled by adding rotated and scaled samples in the training data manually. It also requires a lot of training data to be effective.

Notwithstanding, we got a good impression of how CNNs models can be used in image recognition which has colossal applications in the everyday lives of humans.

5 Bibliography

[1] "CIFAR-10 and CIFAR-100 datasets." https://www.cs.toronto.edu/~kriz/cifar.html (accessed Dec. 25, 2022).

[2] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks." arXiv, Dec. 02, 2015. Accessed: Dec. 26, 2022. [Online]. Available: http://arxiv.org/abs/1511.08458

[3] V. Biscione and J. Bowers, "Learning Translation Invariance in CNNs." arXiv, Nov. 06, 2020. Accessed: Dec. 26, 2022. [Online]. Available: http://arxiv.org/abs/2011.11757

[4] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, Aug. 2017, pp. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

[5] P. Singh and A. Manure, "Neural Networks and Deep Learning with TensorFlow," in *Learn TensorFlow 2.0: Implement Machine Learning and Deep Learning Models with Python*, P. Singh and A. Manure, Eds. Berkeley, CA: Apress, 2020, pp. 53–74. doi: 10.1007/978-1-4842-5558-2_3.