Convolutional Neural Networks and CIFAR-10

Jackson

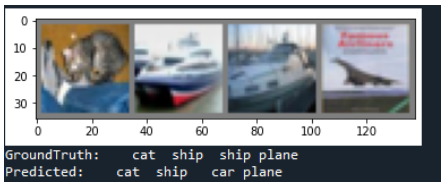
README ON GITHUB

Convolutional neural networks (CNNs) are class of deep learning frameworks primarily designed to work with image data. The core of a CNN is the convolution layer that performs the convolution operation. In mathematics, convolution is an operation that computes the element-wise dot product of an input matrix and a vector. In a CNN, the input matrix is the two-dimensional image while the vector (filter) is the set of weights. Convolutional neural networks gain their power due to translation invariance (Bengio et al. 341). In other words, if a filter is designed to extract or detect a specific feature in an image, the systematic application of the filter across the image will facilitate the discovery of said feature anywhere in the image. This paper is a report on a project to build an image recognition model using the CIFAR-10 dataset. The focus, however, will be on the methods to improve the accuracy of the machine learning model.

CIFAR-10 Dataset

The Canadian Institute for Advanced Research (CIFAR-10) is a standard dataset used in computer vision and deep learning to build classification models. The dataset is especially important when learning how to develop, train, evaluate, and implement deep learning neural networks for object recognition. The dataset contains 60,000 color photographs with ten object classes like birds, cats, dogs, airplanes, frogs, and among other possible image sets. Each image has the size 32x32 pixels while there are 6000 unique images for every class (Recht et al. 4). When training a convolutional neural network, it is recommended that 50,000 images

be used to train the network and the remaining 10,000 images be used as testing images for evaluating the accuracy of the network.



Case Study: Benchmark Convolutional Neural Network

A simple convolutional neural network was written in python with two convolutional layers. The model was trained using 10,000 images from the CIFAR-10 dataset with a constant learning rate of 0.001. However, the epoch and batch sizes were varied. Furthermore, two activation functions (ReLU and sigmoid activation) were used. The resultant network had an accuracy of 62%. According to Recht et al (4), a standard CNN trained with the CIFAR-10 dataset has a benchmark accuracy of over 80%. Therefore, it is essential to discuss ways to improve the accuracy of the network built in this project. In my own testing I achieved an accuracy of 64% percent during testing using the CIFAR-10 dataset. I found that modifying both the epoch and batch size increased accuracies of the resultant networks. Increasing them returned a lower loss.

Discussion

In machine learning, overfitting is a problem where a model performs better with the training dataset (higher accuracy) than with the testing dataset. According to Chen et al. (6232), overfitting happens when the model is too complex for the problem it is being used to solve. In other words, an overfitted convolutional neural network will tend to memorize the

training dataset, thus having a higher accuracy. However, when evaluated against new data (the testing dataset), it will perform poorly. There are numerous solutions to solving the overfitting problem. It should be noted that the accuracy of a convolutional neural network can also be affected by its architecture, such as the number of convolutional layers. However, assuming the architecture of the network is fixed, the following methods are applicable.

Regularization

Regularization is an approach to increase the accuracy of a convolutional neural network by simplifying it. While regularization is an applicable technique, it is not the most optimal. The following excerpt from the paper *Dropout: A simple Way to Prevent Neural Networks from Overfitting* explains why.

> With unlimited computation, the best way to "regularize" a fixed-sized model is to average the predictions of all possible settings of the parameters, weighting each setting by its posterior probability given the training data (Srivastava et al. 1929).

The only challenge is that the training and tuning time for the networks increases exponentially with the size and complexity of the convolutional neural network. Therefore, an alternative solution is to dropout nodes in the deep neural network.

Dropout regularization

Dropout regularization is a simple solution to the overfitting problem, where the neural network will drop nodes in a random manner during training. The outcome is that the nodes will not participate in the training process. As a result, the remaining nodes will be forced to compensate. The overall outcome is that the network will be less likely to memorize the training dataset and instead learn the generalizable features. The dropout procedure can be

implemented for each layer, where an additional dropout layer is added. The number of nodes dropped out for every iteration can be controlled by a single parameter, thus reducing the complexity of the operation. Furthermore, dropout regularization can be implemented using different patterns, such as giving the programmer the freedom to choose where to add the dropout layers, how much dropout to use, and even determine if the dropout should be fixed or dynamic. It should be noted that dropout regularization is a stochastic procedure. Therefore, the accuracy of the trained model will be non-deterministic.

Data Augmentation

An alternative to dropout regularization is data augmentation, where the programmer makes copies some of the elements in the training datasets then applies small random modification. The outcome is that the network both has a larger training dataset while it is being forced to learn the general features of the training set. Data augmentation can be applied in many ways, such as horizontal flip, cropping the image then zooming the resulting low-quality image, and applying minor shifts. However, care should be taken when adding ransom augmentations to copies of the training dataset. If the modifications are too much, the image loses some of the identifiable features and the accuracy of the resultant network will drop instead. Similar to dropout regularization, data augmentation is stochastic. Therefore, the results will be non-deterministic. In other words, when training different neural networks, the outputs will be different. However, the law of large numbers dictates that if the neural network is trained with a large enough dataset, the performance parameters like accuracy will converge to a single number. The only problem is that such a system will require considerable computational resources.

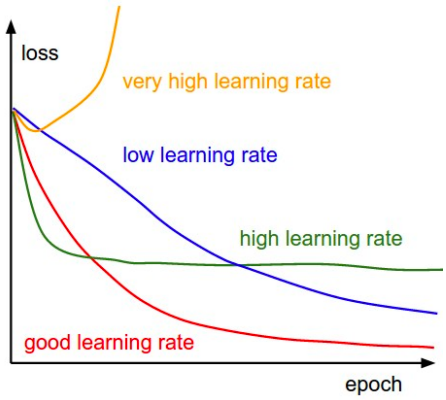
Other Solutions

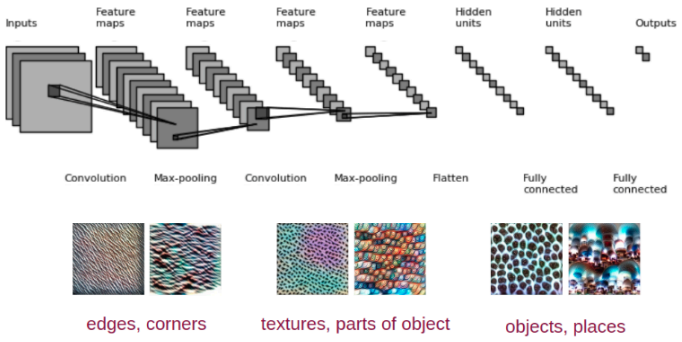
Data augmentation and dropout regularization are not the only solutions to increasing the accuracy of a convolutional neural network. Early stopping is an alternative, where the network is trained to the point where it fits the training dataset but does not start to memorize (overfit). In other words, when the accuracy of the of the model starts to converge towards the training dataset and diverges from the test dataset, early stopping can be applied. Alternatively, the programmer can augment the learning rate, where the schedule drops when the changes in the test accuracy converge to a constant value.

Conclusion

This paper is a report on a project to build an image recognition model using the CIFAR-10 dataset. A simple convolutional neural network was written in python with two convolutional layers. The model was trained using 10,000 images from the CIFAR-10 dataset with a constant learning rate of 0.001. The resultant network had an accuracy of 62%. Different approaches on how to improve the accuracy of the network were discussed. For instance, dropout regularization could be used, where the neural network drops nodes in a random manner during training. The overall outcome is that the network will be less likely to memorize the training dataset and instead learn the generalizable features. Alternatively, data augmentation could be used, where the programmer makes copies some of the elements in the training datasets then applies small random modification. The outcome is that the network both has a larger training dataset while it is being forced to learn the general features of the training set. Regardless of the method used, the accuracy of the resultant network could be improved.

```
Accuracy of the network on the 10000 test images: 62 %
```


loss
very high learning rate
low learning rate
high learning rate
good learning rate
epoch


Inputs
Feature maps
Feature maps
Feature maps
Feature maps
Hidden units
Hidden units
Outputs
Convolution
Max-pooling
Convolution
Max-pooling
Flatten
Fully connected
Fully connected
edges, corners
textures, parts of object
objects, places

Works Cited

Abouelnaga, Yehya, et al. "Cifar-10: Knn-based ensemble of classifiers." *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, 2016.

Bengio, Yoshua, Ian Goodfellow, and Aaron Courville. *Deep learning*. Vol. 1. Massachusetts, USA:: MIT press, 2017.

Chen, Yushi, et al. "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks." *IEEE Transactions on Geoscience and Remote Sensing* 54.10 (2016): 6232-6251.

Recht, Benjamin, et al. "Do cifar-10 classifiers generalize to cifar-10?." *arXiv preprint arXiv: 1806.00451* (2018).

Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* 15.1 (2014): 1929-1958.