Task 1: Identifying training problems of a deep CNN

In-Depth CNN Analysis

A convolutional neural network, often known as a ConvNet or a CNN, is a kind of neural network that incorporates neurons and whose parameters, such as weights, may be trained. The multiple inputs that each neuron takes in are weighted, and the activation function is applied to the resultant signal to produce an output.

Layers of convolutional neural networks

The ConvNet has a layer called a Convolutional Layer, which is responsible for extracting features (such as edges, corners, and endpoints) from the input picture [1]. Convolution is the multiplication of a collection of matrices with the output of the preceding layer.

Drawbacks

The development of a ConvNet that is capable of object recognition on par with that of humans has proved to be a challenging task.

Any part of the picture may be used by a well-trained ConvNet to locate the item being sought. However, the ConvNet will struggle to recognize the item in the picture if it is subject to rotations and scaling [2].

While classifying photos that are very similar to those in the dataset, Convolutional Neural Networks (CNNs) do exceptionally well. CNNs often have trouble with image classification if the pictures are tilted or rotated (refer to Figure below). Data Augmentation, which involves adding new data to a picture during training, is one way to address this issue.

CIFAR 100, perhaps the most well-known picture dataset (14 million photos, 200 classes, 500 images per class) , has been shown to be inadequate for computer vision training since it only includes photographs captured under perfect lighting and angles, rather than the full range of possible inputs.

Convolutional neural networks (CNNs) are classified in depth. The ConvNet's input-close layers aid in the classification of basic characteristics like edges, corners, endpoints, etc. However, layers existing at a deeper level are responsible for transforming basic traits into complex ones. Ultimately, the deepest layer synthesizes all the inputs into a single forecast.

Almost identical to a CNN's eye, the two images above depict essentially the same scene. There may or may not be a face shown in the picture on the right, despite the presence of two eyes, a nose, and a mouth.

The significance of this resides, therefore, in knowing the precise location of the image's objects, which the CNN cannot recognize.

A technique called "data augmentation" could help with this issue. The method of data augmentation often centers around a step in which the picture is flipped or rotated by very little amounts in order to train the dataset. Thus, the CNN learns from a variety of examples [3].

However, complicated pixel manipulation in real-world settings, such as a crumpled T-shirt or an inverted chair, means that data augmentation doesn't handle the worst-case scenario.

Data Enrichment:

The abundance of modern data sources is largely responsible for Deep Learning's recent success. As more data is trained into it, its performance steadily rises. while most other machine learning methods appear to have reached a wall.

Negative aspects of CNN

- For example, the maxpool operation greatly slows down a 2A Convolutional neural network.
- If the CNN contains several layers, training will take a long time on a machine without a powerful graphics processing unit (GPU).
- The processing and training of a ConvNet neural network needs a large dataset.

Instance of the Model

The activation function specifies the boolean value of a binary class matrix, which may be either False or True. The classifier may attempt a linear fit to the integers 0-9 in the picture if we utilized them. For the labels, we may instead generate a 10-element vector that contains a one at the coordinates of the corresponding number in the picture and zeros everywhere else .

Standardization of Multiple Items in a Batch

The technique of normalization, a kind of data preprocessing, is used to convert the raw numbers into a consistent scale without altering their original form.

Training times may be reduced with the use of a method called batch normalization, which also offers some regularization [4]. The issue known as the "internal covariate shift problem" is what batch normalization is trying to address in an effort to find a solution. When training a neural network, this issue appears during the deepest layer's training . The model assumes stable weights in preceding layers while updating layer weights. Altering the input distribution to the following layer, which in turn affects

the output distribution of the next layer, and so on, is a common consequence of updating lower-level layers.
In most cases, we normalize the variables before feeding them into a machine learning or a traditional method. We normalize in part to guarantee that our model will generalize well.

To recap, Batch normalization is a technique for improving the performance and consistency of neural networks by adding hidden layers. If a layer receives data as input from a lower layer, the new layer will perform any necessary normalization or standardization.

Training Time

Batch normalization is a method for training that speeds things up by normalizing the activation of the hidden layer.

Deals with Changes in Internal Covariates

The issue of covariate shift within the sample is addressed [5]. This ensures that the input for each layer has the same central tendency and variance. Examine the following case study to learn more about internal covariate shifts.

Covariance structure modification on the inside

To illustrate, let's pretend we're training a model to determine if a given picture depicts a dog or not. Say we only had pictures of white dogs; even those will follow a particular pattern of dispersion. The photographs will be used to modify the model's settings [6].

Modifies the Loss Function to Be More Gradual

By adjusting the model's parameters, batch normalization boosts the training speed by smoothing the loss function.

References

[1] D. O. P. A. N. E. O. a. K. M. E. Arazo, Pseudo-labeling and confirmation bias in deep semisupervised learning, 2019.

[2] C. Z. A. K. J. L. a. A. A. J. T. Ash, Deep batch active learning by diverse, uncertain gradient lower bounds, 2019.

[3] X. Z. W. L. a. S. G. Y. Chen, Semi-supervised learning under class distribution mismatch, 2020.

[4] Y. G. a. Y. Bengio, Semi-supervised learning by entropy minimization, 2005.

[5] X. Z. S. R. a. J. S. K. He, Deep residual learning for image recognition, 2016.

[6] N. C. I. G. N. P. A. O. a. C. A. R. D. Berthelot, Mixmatch: A holistic approach to semisupervised learning, 2019.