Task: MNIST / CIFAR 10 classification using neural networks

Using the MNIST hand written dataset or the CIFAR-10 dataset, build a model for classification of digits from 0-9 using a simple neural network. This task emphasizes your ability to work with deep neural network and training large models using frameworks like Tensorflow or PyTorch.

Import the MNIST dataset or CIFAR-10 dataset.

Pre-process the dataset according to your requirements.

Use data partitioning to generate training set, validation set and test set.

Build a simple CNN using at least 2 layers and add regularization if necessary.

Using an appropriate optimizer evaluate the loss and set a learning rate.

Plot the training and validation loss curve.

Evaluate the model on test set.

Explain the use of batch normalization layer and drop out layer in a neural network. Does your model require these layers? How did it affect the performance of your model these layers? How did it affect the performance of your model.

SOLUTION:

To perform the above-mentioned task and see how accurately we can classify the images in CIFAR10 dataset. For this task, I have tried to check the difference between the accuracy of one CNN model with different parameters. The accuracy measure and the training-validation curve help us understand which model can help us with better predictions and how each of these is different in a few terms mentioned below:

Base CNN Model:

The model achieves a test accuracy of 67%. This model consists of 3 convolutional layers, without dropout and batch normalization. The model has few degrees of freedom, and it may even be underfitting due to limited complexity, though it tends to overfit.

Model with Dropout:

The model obtained 76% test accuracy. Dropout helps the model's regularization, prevents it from overfitting, and enhances its generalization by randomly turning off some neurons during training.

Model with Batch Normalization and Dropout:

78% on test. Using batch normalization and dropout together stabilizes training, accelerates convergence, and minimizes the chances of overfitting.

**Explanation of Batch Normalization and Dropout Layers**

1. Batch Normalization Layer:

Objective: Batch Normalization normalizes inputs to each layer, so every layer receives input data which most often has a consistent distribution with mean close to 0 and variance close to 1. This ensures that the inner covariate shift, so actually trains faster more stably, and potentially use higher learning rates.

Effect on Performance:

* Training Stability: Batch normalization helps in stabilizing the training process, resulting in smoother and faster convergence.
* Generalizes better- it minimizes the likelihood of overfitting because it helps reduce the network's sensitivity to weight initialization and changes of input distributions.

2. Dropout Layer:

Objective: Dropout is a form of regularization where neurons are randomly selected to be ignored during the training of the model, hence termed dropout. This prevents the model from relying too heavily on specific neurons and encourages the network to learn more robust features.

Effect on Performance:

* Overfitting Prevention : This is just how dropout prevents any particular neuron from over-representing the predictions and forces the model to learn the patterns in a more generalized way.

**Does Your Model Require These Layers?**

Without Dropout and Batch Normalization: The same base CNN model presumably had either overfitting or underfitting. This means the addition of more regularization or better learning stability would lead to further improvements.

Only Dropout: Accuracy increased to 76%, so we see that overfitting was removed, and there was slightly better generalization.

To these, the highest accuracy was 78% when Batch Normalization and Dropout were applied to the model. The batch normalization likely gave the model stable training that dropout would have imparted and regularization that dropout would have otherwise given back in return.

**In What Way Did It Affect the Performance of Your Model?**

Improved Convergence: The batch normalization helped bring faster convergence and smoother convergence with less oscillation in the loss curve.

More importantly, dropout did reduce overfitting and improve validation accuracy whereas batch normalization ensured stable training.

Test accuracy: Adding both layers yielded the highest test accuracy; this overall suggests that the model is improved over both prior regularization- and stable-training approaches.

In short, it added both batch normalization and dropout, which greatly improved the robustness of training stability reduced overfitting capabilities, and better generalization capabilities as indicated through better test accuracy.

The link for the Kaggle notebook is added for your reference:

Kaggle link: <https://www.kaggle.com/code/kanishkajuneja/cifar10>