ID: 22308011
Name: Ajoy Shutra Dhar

Assignment on Deep Learning

Part(A): Theoretical Concepts:

1. Explain Activation Function:

Simply put, an artificial neuron calculates its inputs ‘ weighted sum’ and adds a bias, as shown in the figure below by the net input.

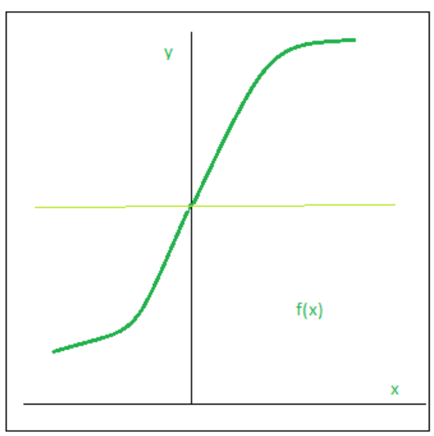
Net Input = $\sum($ Weight $\times$ Input$)$ Bias

The value of net input can be anything from- inf to inf. The neuron doesn’t know how to bound to value and therefore is not set up to decide the blasting pattern. The activation function is an important part of an artificial neural network. They decide whether a neuron should be actuated or not. However, it bounds the value of the net input. The activation function is an anon-direct metamorphosis that we do over the input before transferring it to the coming estate of neurons or finishing it as an affair.

Define and compare the activation Function given below:

- ☐ **Sigmoid Function**: Sigmoid function is a widely used activation function. It is defined as:

Formula: $\sigma(x) = 1/(1 + e^x)$



This is a smooth function and is continuously differentiable. The biggest advantage that it has over step and direct function is that nonlinear. This is an incredibly cool point of the sigmoid function. This means that when I've multiple neurons having the sigmoid function as their activation function – the affair is nondirect as well. The function ranges from 0- 1 having an S shape.

Use Case: Binary classification, Linear regression
Limitations: The network does not learn rapidly close to the borders, which is a drawback of the sigmoid function.

- ☐ ReLU(remedied direct unit) In deep literacy, one of the most popular activation functions is the remedied Linear Unit(ReLU). It gives a model of non-linearity, which aids in the literacy of intricate data patterns.

Formula: f(x)=max(0,x)
These mean,

- f(x)>0; f(x) =0 x;
- f(x)<=0; f(x) ==0;

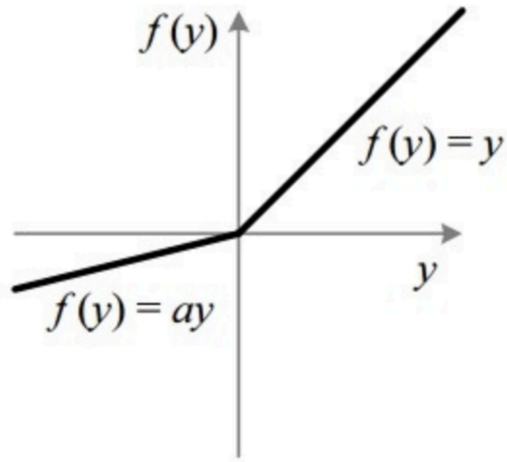
Limitations:

- Unbounded Affair ReLU laborers have the eventuality to become unstable due to their measureless growth.

Use Case

- Generally used in retired layers of deep literacy models, especially CNNs and MLPs.
- Suitable for tasks where meager activation is salutary.

- ☐ **Leaky ReLU:** Leaky remedied Linear Unit or Leaky ReLU, is a type of activation function grounded on a ReLU, but it has a small pitch for negative values rather than a flat pitch. The pitch measure is determined before training, i.e. it isn't learned during training. This type of activation function is popular in tasks where we may suffer from meager slants, for illustration training generative inimical networks.



- **Use Case:** Addresses dying ReLU.
- **Limitation**: Requires additional parameter tuning.

☐ **Tanh:** Formula :
$f(x) = e^x - e^{-x}/e^x + e^{-x}$
Historically, the tanh function was preferred over the sigmoid function as it gave better performance in multi-layer neural networks. But it didn't break the evaporating grade problem that sigmoids suffered, which was dived more effectively with the preface of ReLU activations.

Use Case: The tanh function is constantly used in the retired layers of a neural network. Because of its zero-centered nature, when the data is also regularized to have to mean zero, it can affect more effective training.

Limitations: All the negative values become zero immediately

2. <u>Discuss Optimization Algorithm</u>

SGD(Stochastic Gradient Descent)

- Advantage - Simple and memory-effective.
- Limitation - Slow confluence.

Adam

- Advantage - Combines instigation and adaptive literacy rate.
- Limitation - Sensitive to hyperparameters.

RMSprop

- Advantage- Good for non-stationary objects.

- Limitation- May overfit without proper tuning.
- Learning Rate - Affects confluence speed and stability. Dynamic adaptation through schedulers improves training.

☐ Error Analysis

Common Errors

- Misclassification of analogous classes(e.g., cat vs. canine).
- Poor performance in lower frequent classes.
- perceptivity to gyration or other accruals.

Solutions:

- Increase training data for inadequately performing classes.
- Use more advanced accruals(e.g., brilliance adaptation).
- Fine-tune the model or try transfer literacy.

Problem understanding:

Build a deep neural network for image classification using the CIFAR-10 dataset. perform normalization and data augmentation techniques, design CNN architecture with at least 3 convolutional layers and 2 fully connected layers, and include dropout and batch normalization to regularize the model. Train the model and display the training and validation accuracy /loss curve. use early stopping if necessary. Evaluate the model on test set and report the classification accuracy.

Visualize a confusion matrix and do an error analysis :

- a)identify and discuss three common types of errors your model makes (e.g. specific classes frequently misclassified)
- b)Suggest potential solutions to reduce these errors

☐ Model Design

The CIFAR-10 dataset contains 60,000 color images across 10 classes. The objective is to classify these images using a Convolutional Neural Network (CNN). The model follows a sequential architecture with the layers and hyperparameters:

- **Input Layer**: The input layer is d using the `Input` class for images of shape (32, 32, 3).
- **Convolutional Layers**: Multiple convolutional layers occupying ReLU activation extract spatial features. Kernel sizes vary, having a consistent stride of 1.
- ·**Pooling Layers**: MaxPooling layers reduce the spatial dimensions, preventing overfitting
- ·**Dropout Layers**: Regularization is gained using dropout layers to reduce overfitting.

- ·**Fully Connected Layers**: Dense layers process the extracted features into class probabilities.
- **Output Layer**: using a final dense layer with a `softmax` activation function outputs probabilities for each of the 10 classes.
- ·**Optimization and Loss Function**: The model uses the Adam optimizer with a learning rate of $1\times10-3$ and categorical cross-entropy for the loss function.

The model is trained using a three-fold cross-validation approach to ensure robustness and generalization across unseen data.

☐ Result and Discussion

Training and Validation Accuracy

The model was trained for 50 epochs, and performance was evaluated on both training and validation datasets. Here are given the key observations:

- **Training Phase:** The accuracy steadily improved across epochs, reaching ~80% on the training data by the 50th epoch.
- **Validation Phase:** Validation accuracy varies across folds, achieving the highest value of 78.10% in fold 1. However, performance across other folds demonstrated inconsistencies, with validation accuracy dropping to 27.49% in fold 2 and 32.83% in fold 3.

Test Results

Testing on the held-out dataset yielded an accuracy of 80.85%, reflecting reasonable generalization. The classification report provided detailed insights into class-wise performance:

- · Precision and recall class, with the best results observed for classes such as 1 (automobile) and 8 (ship), achieving F1-scores of 0.89 and 0.87.
- · Lower performance was observed for classes 0 (airplane) and 3 (cat), where F1-scores dipped to 0.79 and 0.70, indicating difficulty in making a difference between these categories due to their high visual similarity to other classes.

Challenges and Observations

- Overfitting in specific folds regularization techniques like data augmentation or stronger dropout could help.
- Classes with lower accuracy might benefit from reprocessing or the inclusion of more advanced techniques, such as transfer learning using pre-trained models.

☐ Conclusion

This project demonstrates the development and evaluation of a CNN for CIFAR-10 image classification. While the model gained good results with a test accuracy of 80.85%, some inconsistencies in fold performance suggest improvement in regularization and model tuning.

Future Directions:

1. Incorporate advanced architectures, such as ResNet or EfficientNet, mobilenet, and VGG16, to improve performance.
2. Data augmentation strategies to increase dataset diversity.
3. Experiment with hyperparameter optimization for better performance.