Working_of_deep_neural_network_for_classificaion_p2

October 22, 2025

Write a program to demonstrate the working of a deep neural network for classification task.

[2]:
```
# Deep Neural Network for Classification
import torch
import torch.nn as nn
import torch.optim as optim
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# 1. Load and prepare data
iris = load_iris()
X = iris.data
y = iris.target

X = StandardScaler().fit_transform(X)  # normalize
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

X_train = torch.tensor(X_train, dtype=torch.float32)
y_train = torch.tensor(y_train, dtype=torch.long)
X_test  = torch.tensor(X_test, dtype=torch.float32)
y_test  = torch.tensor(y_test, dtype=torch.long)

# 2. Define the DNN model
class DNN(nn.Module):
    def __init__(self):
        super().__init__()
        self.net = nn.Sequential(
            nn.Linear(4, 8), nn.ReLU(),
            nn.Linear(8, 6), nn.ReLU(),
            nn.Linear(6, 3)  # 3 output classes
        )
    def forward(self, x): return self.net(x)

model = DNN()
optimizer = optim.Adam(model.parameters(), lr=0.01)
loss_fn = nn.CrossEntropyLoss()
```

```python
# 3. Train the model
for epoch in range(200):
    optimizer.zero_grad()
    y_pred = model(X_train)
    loss = loss_fn(y_pred, y_train)
    loss.backward()
    optimizer.step()
    if epoch % 50 == 0:
        print(f"Epoch {epoch}, Loss={loss.item():.4f}")

# 4. Evaluate
with torch.no_grad():
    preds = torch.argmax(model(X_test), dim=1)
    acc = (preds == y_test).float().mean()
    print(f"Accuracy: {acc*100:.2f}%")
```

```
Epoch 0, Loss=1.1563
Epoch 50, Loss=0.2293
Epoch 100, Loss=0.0475
Epoch 150, Loss=0.0364
Accuracy: 96.67%
```

[ ]:
```
s
```