Explanation of Training Parent CNN

This section explains the code used to train the ParentCNN model on the MNIST dataset.

```
tree = NetworkTree()
parent_model = ParentCNN().to(device)
cirterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(parent_model.parameters(), lr=0.001)
```

- tree: Creates an instance of NetworkTree to manage models and data.
- parent_model: Instantiates the ParentCNN model and moves it to the selected device (CPU or GPU).
- cirterion: (typo, should be 'criterion') Sets the loss function to cross-entropy, suitable for classification.
- optimizer: Sets up the Adam optimizer for the model's parameters with a learning rate of 0.001.

```
print("Training Parent CNN on MNIST data...")
for epoch in range(2):
    parent_model.train()
    for data, target in mnist_loader:
        data, target = data.to(device), target.to(device)
        optimizer.zero_grad()
        output = parent_model(data)
        loss = criterion(output, target)
        loss.backward()
        optimizer.step()
```

```
    print(f"Epoch {epoch+1}, Loss: {loss.item():.4f}")
```

- The code trains the model for 2 epochs. For each batch:
  - Sets the model to training mode.
  - Moves data and labels to the device.
  - Clears previous gradients.
  - Computes model output and loss.
  - Backpropagates the loss and updates model weights.
  - Prints the loss for each epoch.

```
tree.add_model("parent", parent_model, "MNIST", {"loader": mnist_test_loader, "sample": next(iter(mnist_loader))[0], "type": "image"})
```

- Adds the trained parent model to the tree with its associated data and type.

Purpose:

- This block trains the main CNN model on MNIST and registers it in the model management tree for later use or transfer.