Model Architecture

November 23, 2023

1 Model Layers

The model architecture consists of the following layers:

```
model = nn.Sequential(
            nn.Conv2d(1, 64, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(64, 128, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Flatten(),
            nn.Linear(128 * 7 * 7, 128),
            nn.ReLU(),
            nn.BatchNorm1d(128, self.config['model_norm']),
            nn.Linear(128, 64),
            nn.ReLU(),
            nn.BatchNorm1d(64, self.config['model_norm']),
            nn.Linear(64, 10)
        )
```

2 Forward Function

The forward function of the model is implicitly defined by the `nn.Sequential` container. The forward pass is executed by calling the model instance with input data. For example:

$$\text{outputs} = \text{model}(\text{inputs})$$

3 Gradient Flow

During the training process, the gradient flow is established through the model's layers. The backward pass computes gradients using backpropagation. The optimizer uses these gradients to update the model's weights. Optimizers used [Adam, Adagrad, SGD, RMSProp].

4 Layers contributions to results

- Convolutional layers capture hierarchical features in the input
- Fully connected layers provide non-linear mappings
- Batch normalization helps stabilize and speed up training
- ReLU activations introduce non-linearity
- max-pooling layers downsample the spatial dimensions