Training a Diffusion Model

```python
import torch
import torch.nn as nn
from torchvision.datasets import CIFAR10
from torch.utils.data import DataLoader
from unet import UNet  # predefined U-Net

# Load CIFAR-10 dataset (32x32 color images)
dataset = CIFAR10(root='./data', train=True, download=True,
                  transform=lambda x: torch.tensor(x).float() / 255)
dataloader = DataLoader(dataset, batch_size=32, shuffle=True)

# A model that takes in a noisy image and a timestep
model = UNet(in_channels=4, out_channels=3)  # 3 color + 1 timestep
    channel
optimizer = torch.optim.Adam(model.parameters(), lr=1e-4)
criterion = nn.MSELoss()


num_noise_levels = 1000  # number of noise levels

for epoch in range(100):
    for images, _ in dataloader:  # ignore labels
        # Pick a random noise level for each image in the batch
        t = torch.randint(0, num_noise_levels, (images.shape[0],))
        noise_amount = (t.float() / num_noise_levels).view(-1, 1, 1, 1)

        # Create noisy images
        noise = torch.randn_like(images)
        noisy = (1 - noise_amount) * images + noise_amount * noise

        # Append t as an extra channel (it has shape (batch_size),
        # so we need to broadcast over the spatial dimensions)
        t_channel = noise_amount.expand_as(images[:, :1, :, :])
        model_input = torch.cat([noisy, t_channel], dim=1)

        prediction = model(model_input)
        loss = criterion(prediction, noise_amount * noise)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
```

Questions

In groups, study the code on the previous page. Be prepared to share your ideas with the class.

1. What are the inputs to the model? Describe each channel.

2. Describe the variable `noisy`. When `noise_amount` is close to 0, what does `noisy` look like? When it's close to 1?

3. What is the model being trained to predict?

4. Therefore, what "task" is this model learning to perform?

5. Why does the model need to know `t`?

6. After training, how could you use this model to *generate a new image from scratch*? Sketch an algorithm.