Variational Autoencoder

As part of this activity we will add the variational autoencoder to the helper library we developed as part of Module 4.

You will encorporate appropriate code from Module5-VAE notebook into your helper_lib module.

---

1. Model Module

In `model.py`, define your neural network model:

```
import torch.nn as nn

def get_model(model_name):
    # TODO: define and return the appropriate model_name - one of: FCNN, CNN,
EnhancedCNN, VAE
    return model
```

---

2. Trainer Module

In `trainer.py`, add train_vae_model function:

```
import torch

def train_vae_model(model, data_loader, criterion, optimizer, device='cpu',
epochs=10):
    # TODO: run several iterations of the training loop (based on epochs
parameter) and return the model
    return model
```

3. Image Generator

In `generator.py`, add train_vae_model function:

```
import torch
```

```
def generate_samples(model, device, num_samples=10):
    # TODO: generate num_samples points in the latent space, run the decoder to
construct an image, and plot the samples on a grid
    plt.show()
```

---

3. Usage Example

Here's how to use your helper library:

```
from helper_lib.data_loader import get_data_loader
from helper_lib.trainer import train_model
from helper_lib.model import get_model
import torch.nn as nn
import torch.optim as optim

train_loader = get_data_loader('data/train', batch_size=64)
test_loader = get_data_loader('data/test', batch_size=64, train=False)

vae = get_model("VAE")
criterion = # TODO: add a loss function
optimizer = optim.Adam(vae.parameters(), lr=0.001)

train_vae_model(vae, train_loader, criterion, optimizer, epochs=5)
generate_samples(vae, device, num_samples=10)
```