Ramakrishna Mission Vivekananda Educational & Research Institute

Belur Math, Howrah, West Bengal

School of Mathematical Sciences, Department of Computer Science

Assignment-3: Autoencoders, VAEs, and Fourier Transforms for Image Reconstruction & Inpainting.
M.Sc. Computer Science and Big Data Analytics Date: 12 March 2025
Course : **CS411: Applications of Computer Vision and Deep Learning**
Deadline: 12th-April-2025, 11:59 P.M.
Instructor: Jimut Bahan Pal Max marks: 200

Instructions: **Attempt all the questions**; **One single PDF should be submitted** alongside the **jupyter-notebook**. Submit the code and the PDF writings as **Name_ROLL.zip** and send it to the reply email on **jimutbahanpal@yahoo.com** AND **jpal.cs@gm.rkmvu.ac.in** as Carbon Copy (CC). You may also CC it to your other personal email address to check whether the email has gone to the address. **Submission after the deadline will fetch you -5 marks per day, so start early!**. You may discuss the logic and approach with your peers, but you must write and submit your own code. Copying code from others will result in penalties.

To ensure consistency in your machine learning pipeline, prepare all data transformations before beginning the training process:

1. Pre-generate all Fourier splits for your dataset
2. Create all masked image splits in advance
3. Save these prepared splits in the designated folder
4. Apply a consistent random seed of 42 to all non-deterministic components of your pipeline

This approach will maintain data format integrity and ensure reproducibility across runs.

Dataset: **CIFAR-10**. CIFAR-10 is a dataset consisting of 60,000 32x32 color images in 10 classes, with 6,000 images per class. The dataset is divided into 50,000 training images and 10,000 test images. For this assignment, we will further split the training set into 70% training, 30% validation, and keep the test set aside, until asked to evaluate.

1. (a) Simple Autoencoder (200)

    1. **Build and Train a Simple Autoencoder** **(25 Marks)**

        (a) Dataset Preparation: **(5 Marks)**

        - Split the CIFAR-10 dataset into 70% training, 10% validation, and 20% test sets. **(3 Marks)**
        - Normalize the images to the range [0, 1]. **(2 Marks)**

        > Tip
        >
        > Consider using `torchvision.transforms` or `sklearn` for consistent normalization.

        (b) Model Architecture: **(8 Marks)**

        - Build a simple autoencoder with separate encoder and decoder modules. Show the torchsummary and the graphviz vizualization of the model. **(3 Marks)**
        - The encoder should reduce the input image ($32\times32\times3$) to a latent vector of size $n\times1\times1\times1$, where n is a hyperparameter of your choice (e.g., 4, 8, 16, etc.). **(3 Marks)**

- The decoder should reconstruct the image from the latent vector. **(2 Marks)**

> Tip
>
> Consider using convolutional layers with appropriate stride and padding for dimensionality reduction rather than pooling layers.

(c) Training: **(7 Marks)**

- Train the autoencoder on the training set using Mean Squared Error (MSE) as the loss function. **(2 Marks)**
- Use the Adam optimizer with a learning rate of your choice. **(2 Marks)**
- Report the training, validation, and test MSE losses. **(1 Mark)**
- Plot the training and validation loss curves. **(2 Marks)**

> Tip
>
> Use early stopping based on validation loss to prevent overfitting, or do whatever that suits you.

(d) Evaluation: **(5 Marks)**

- Evaluate the model on the test set using the following metrics:
  - Mean Squared Error (MSE) **(1 Mark)**
  - Peak Signal-to-Noise Ratio (PSNR) **(2 Marks)**
  - Fréchet Inception Distance (FID) **(2 Marks)**

> Tip
>
> For FID calculation, consider using a pre-trained model like Inception-v3 for feature extraction.

2. **Latent Space Visualization and Inpainting** **(30 Marks)**

(a) Latent Space Visualization: **(10 Marks)**

- Use the trained encoder to extract latent vectors for all the images from the test set. **(2 Marks)**
- Plot the UMAP (Uniform Manifold Approximation and Projection) of the latent vectors in 2D space colored according to the classes of the images. **(3 Marks)**
- Reconstruct the images using the decoder and compare them to the original images (do this for the first 5 images to show visualization) **(2 Marks)**
- Report the metrics, i.e., MSE, PSNR, and FID for all the images in the test set in mean $\pm$ standard deviation form. **(3 Marks)**

> Tip
>
> When using UMAP, experiment with different `n_neighbors` and `min_dist` parameters to find the best visualization.

(b) Latent Space Manipulation: **(8 Marks)**

- Add minimal random noise to the latent vectors and reconstruct the images. **(3 Marks)**
- Ensure that the visual quality of the reconstructed images does not degrade significantly. **(2 Marks)**
- Show this for the first 5 test images. **(1 Mark)**
- Report the metrics, i.e., MSE, PSNR, and FID for all the test images in mean $\pm$ standard deviation form. **(2 Marks)**

> **Tip**
>
> Try different noise distributions (Gaussian, uniform) and variance levels to see their effect on reconstruction quality.

(c) Inpainting: **(12 Marks)**

- Create a mask of size 8×8 pixels (fully black) and apply it to random locations in the test images, single mask for a single image in a random location. **(3 Marks)**
- Pass the masked images through the encoder to obtain modified latent vectors. **(2 Marks)**
- Plot the UMAP (Uniform Manifold Approximation and Projection) of the latent vectors in 2D space, colored according to the classes of the images for the whole test set. **(2 Marks)**
- Reconstruct the images using the decoder and evaluate the inpainting performance. **(2 Marks)**
- Report the MSE, PSNR, and FID for the inpainted test images. **(2 Marks)**
- Visualize the original, masked, and inpainted images for 5 test samples. **(1 Mark)**

> **Tip**
>
> Consider creating a visualization grid showing the original, masked, and reconstructed images side by side for easy comparison.

(b) Variational Autoencoder (VAE) **(65 Marks)**

1. **Build and Train a Variational Autoencoder** **(25 Marks)**

(a) Model Architecture: **(10 Marks)**

- Modify the autoencoder architecture from Part 1 to create a Variational Autoencoder (VAE). Show the torchsummary and the graphviz vizualization of the model. **(3 Marks)**
- The encoder should output the mean and log-variance of the latent distribution. **(4 Marks)**
- Use the reparameterization trick to sample from the latent distribution. **(3 Marks)**

> **Tip**
>
> Ensure your log-variance outputs are numerically stable by adding a small epsilon or applying activation functions.

(b) Training: **(8 Marks)**

- Train the VAE on the training set using a combination of reconstruction loss (MSE) and KL divergence loss. **(3 Marks)**
- Use the Adam optimizer with a learning rate of your choice. **(2 Marks)**
- Report the training, validation, and test losses. **(1 Mark)**
- Plot the training and validation loss curves. **(2 Marks)**

> **Tip**
>
> Consider using a beta parameter to weight the KL divergence term (β-VAE) for better disentanglement of latent features.

(c) Evaluation: **(7 Marks)**

- Evaluate the model on the test set using the following metrics:
  - Mean Squared Error (MSE) **(2 Marks)**

- Peak Signal-to-Noise Ratio (PSNR) **(2 Marks)**
- Fréchet Inception Distance (FID) **(3 Marks)**

> Tip
>
> Compare these metrics with those from the simple autoencoder to understand the trade-offs between reconstruction quality and generative capabilities.

2. **Latent Space Visualization and Inpainting** **(40 Marks)**

(a) Latent Space Visualization: **(12 Marks)**

- Use the trained VAE encoder to extract latent vectors for 5 images from the test set. **(2 Marks)**
- Use the trained encoder to extract latent vectors for all the images from the test set. **(2 Marks)**
- Plot the UMAP (Uniform Manifold Approximation and Projection) of the latent vectors in 2D space, by taking the final vector from the two vectors, colored according to the classes of the images for the whole test set. **(3 Marks)**
- Reconstruct the images using the decoder and compare them to the original images (do this for the first 5 images to show visualization) **(2 Marks)**
- Report the metrics, i.e., MSE, PSNR, and FID for all the test images in mean $\pm$ standard deviation form. **(3 Marks)**

> Tip
>
> For VAE latent space, try sampling multiple points from the learned distribution for each input to observe the variation in reconstructions.

(b) Latent Space Manipulation: **(10 Marks)**

- Add minimal random noise to the latent vectors and reconstruct the images. **(5 Marks)**
- Ensure that the visual quality of the reconstructed images does not degrade significantly. **(5 Marks)**
- Do this for the first five samples from the test set.

> Tip
>
> Try interpolating between latent vectors of different classes to visualize the continuous nature of the VAE latent space.

(c) Inpainting: **(18 Marks)**

- Apply the same 8×8 mask from Part 1 to the test images. **(2 Marks)**
- Pass the masked images through the VAE encoder to obtain modified latent vectors for the full test set. **(3 Marks)**
- Reconstruct the images using the decoder and evaluate the inpainting performance. **(4 Marks)**
- Report the MSE, PSNR, and FID for the inpainted images. **(3 Marks)**
- Visualize the original, masked, and inpainted images for 5 test samples and report the MSE for the visualization. **(3 Marks)**
- Compare the inpainting performance of the VAE with the simple autoencoder from Part 1. **(3 Marks)**

> **Tip**
>
> The probabilistic nature of VAEs often leads to more natural-looking inpainting. Consider discussing why this might be the case in your analysis.

(c) Fourier Transformations and Autoencoders **(80 Marks)**

1. **Fourier Transform and Reconstruction** **(20 Marks)**

   (a) Fourier Transform: **(12 Marks)**

   - Select an image from the training set and compute its Fourier transform. **(2 Marks)**
   - Convert the Fourier-transformed image back to the spatial domain. **(2 Marks)**
   - Report the MSE, PSNR, and FID between the original and reconstructed images. **(3 Marks)**
   - Visualize the original, Fourier-transformed, and reconstructed images for the first five samples in the test set. **(2 Marks)**
   - Do this for all the images in the test set and report the metrics. **(3 Marks)**

   (b) Noise Addition: **(8 Marks)**

   - Add Gaussian noise to the Fourier-transformed image. **(3 Marks)**
   - Perform the inverse Fourier transform and visualize the reconstructed image for first five samples in the test set. **(2 Marks)**
   - Report the MSE, PSNR, and FID between the original and noisy-reconstructed images for all the test images. **(3 Marks)**

   > **Tip**
   >
   > Try adding noise to different frequency components (high vs. low) to observe the effects on the reconstructed image.

2. **Autoencoder for Fourier Transform Learning** **(30 Marks)**

   (a) Autoencoder-1 (Fourier Transform Learning): **(7 Marks)**

   - Train an autoencoder (Autoencoder-1) to learn the Fourier transform of the train images, hence you need to create the fourier transform of each of the splits, i.e., train, validation and test. **(4 Marks)**
   - The input to Autoencoder-1 is the original image, and the target output is its Fourier transform. **(2 Marks)**
   - Use MSE as the loss function. **(1 Mark)**
   - Report the training, validation and test MSE losses. **(1 Mark)**

   (b) Autoencoder-2 (Inverse Fourier Transform Learning): **(7 Marks)**

   - Train another autoencoder (Autoencoder-2) to learn the inverse Fourier transform. **(3 Marks)**
   - The input to Autoencoder-2 is the Fourier-transformed image, and the target output is the original image. **(2 Marks)**
   - Use MSE as the loss function. **(1 Mark)**
   - Report the training and test MSE losses. **(1 Mark)**
   - You can create the same training pipeline to train these two models or do them separately, if done separately, then please save each of the outputs which will be used, but I recommend to do it in the same pipeline (figure it out yourselves). **(0 Marks, just guidance)**

   (c) Evaluation: **(6 Marks)**

   - Use Autoencoder-1 to generate Fourier-transformed images for the test set. **(1 Mark)**

- Use Autoencoder-2 to reconstruct the original images from the Fourier-transformed images. **(1 Mark)**
- Report the MSE, PSNR, and FID for the reconstructed images of the test set. **(2 Marks)**
- Visualize the original, Fourier-transformed, and reconstructed images for 5 test samples, and show the MSE for these. **(2 Marks)**

> Tip
>
> Create a pipeline where the output of Autoencoder-1 is fed directly into Autoencoder-2 to evaluate end-to-end performance.

(d) Latent Space Clustering: **(5 Marks)**

- Extract latent vectors from Autoencoder-1 and Autoencoder-2 for the test set. **(2 Marks)**
- Plot the UMAP of the latent vectors in 2D space for all the test samples colored according to the classes. **(2 Marks)**
- Compare the clustering patterns of the latent vectors from both autoencoders. **(1 Mark)**

(e) Inpainting with Fourier Transforms: **(5 Marks)**

- Apply the 8×8 mask to the test images and perform inpainting using Autoencoder-1 and Autoencoder-2. Show this for first 5 images from the test set. **(2 Marks)**
- Report the MSE, PSNR, and FID for the inpainted images. **(2 Marks)**
- Compare the inpainting performance with the results from Part 1 and Part 2. **(1 Mark)**

3. **Variational Autoencoder (VAE) for Fourier Transform Learning** **(30 Marks)**

(a) VAE-1 (Fourier Transform Learning): **(7 Marks)**

- Train a VAE (VAE-1) to learn the Fourier transform of the train images. **(3 Marks)**
- Create the Fourier transform for each of the splits (train, validation, test). **(1 Mark)**
- The input to VAE-1 is the original image, and the target output is its Fourier transform. **(2 Marks)**
- Use a combination of reconstruction loss (MSE) and KL divergence loss. **(1 Mark)**
- Report the training and test losses. **(1 Mark)**

(b) VAE-2 (Inverse Fourier Transform Learning): **(7 Marks)**

- Train another VAE (VAE-2) to learn the inverse Fourier transform. **(3 Marks)**
- The input to VAE-2 is the Fourier-transformed image, and the target output is the original image. **(2 Marks)**
- Use a combination of reconstruction loss (MSE) and KL divergence loss. **(1 Mark)**
- Report the training and test losses. **(1 Mark)**
- You can create the same training pipeline to train these two models or do them separately. If done separately, save the outputs for later use, which will be used, but I recommend to do it in the same pipeline (figure it out yourselves). **(0 Marks, just guidance)**

> Tip
>
> Consider using a conditional VAE approach where class information guides the generation process.

(c) Evaluation: **(6 Marks)**

- Use VAE-1 to generate Fourier-transformed images for the test set. **(1 Mark)**

- Use VAE-2 to reconstruct the original images from the Fourier-transformed images. **(1 Mark)**
- Report the MSE, PSNR, and FID for the reconstructed images of the test set. **(2 Marks)**
- Visualize the original, Fourier-transformed, and reconstructed images for 5 test samples, and show the MSE for these. **(2 Marks)**

> Tip
>
> Generate multiple samples from each VAE to showcase the probabilistic nature of the models.

(d) Latent Space Clustering: **(5 Marks)**

- Extract latent vectors from VAE-1 and VAE-2 for the test set. **(2 Marks)**
- Plot the two UMAPs of the latent vectors in 2D space. **(2 Marks)**
- Compare the clustering patterns of the latent vectors from both VAEs. **(1 Mark)**

> Tip
>
> Analyze how the KL divergence regularization affects the structure of the latent space compared to the traditional autoencoders.

(e) Inpainting with Fourier Transforms: **(5 Marks)**

- Apply the 8×8 mask to the test images and perform inpainting using VAE-1 and VAE-2. Show this for first 5 images from the test set. **(2 Marks)**
- Report the MSE, PSNR, and FID for the inpainted images. **(2 Marks)**
- Compare the inpainting performance with the results from Part 1 and Part 2. **(1 Mark)**

> Tip
>
> The probabilistic nature of VAEs can produce multiple plausible reconstructions for masked regions. Consider showcasing this capability, if possible.

Submission Guidelines

- Submit a well-documented Jupyter Notebook or Python script with all the code. **(Required)**
- Include visualizations, plots, and metrics for each task. **(Required)**
- Provide a brief report summarizing your observations and results. **(Required)**
- All code should be properly commented and organized in a logical manner. **(Encouraged)**
- Include a `requirements.txt` file with all the necessary dependencies. **(Encouraged)**
