HOMEWORK 2

A homework report
submitted to **Prof. Sunita Sarawagi**
in the subject of Advanced Machine Learning

by

Prateek Chanda (22D0362)
Sandarbh Yadav (22D0374)

INDIAN INSTITUTE OF TECHNOLOGY BOMBAY

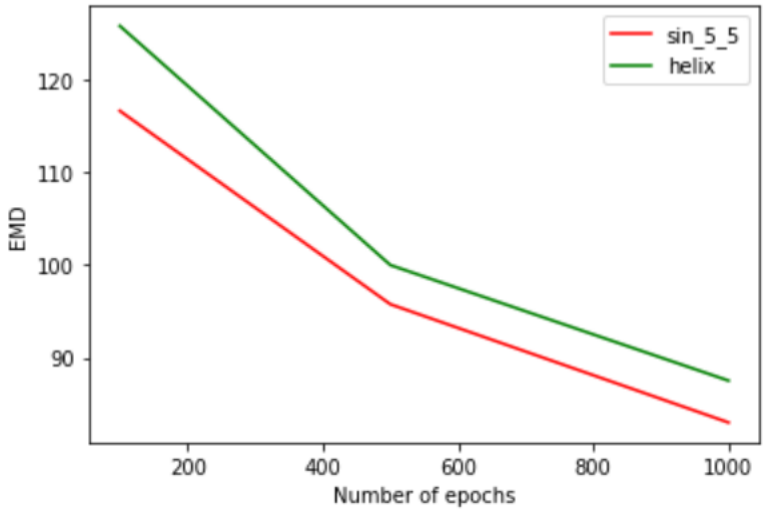
2023

Q 1. **Number of training steps** - *Does training longer help?* - Train at least 3 DDPMs with varying number of epochs (say 500, 1000, 2000 respectively), keeping all other hyperparameters fixed. Produce a sample quality metric vs number of epochs plot in the report along with your comments about presence or absence of a trend in the plot. Don't forget to mention values of all other hyperparameters that you kept fixed and their possible impact on the presence or absence of a trend in your plot.

Answer: The results are reported below:

Number of epochs	EMD (sin_5_5)	EMD (helix)
100	116.59	125.74
500	95.37	99.76
1000	**83.29**	**87.54**

Table 1: EMD values on test set corresponding to number of epochs for both datasets



Clearly on increasing the number of epochs, EMD decreases for both datasets. The explanation for this trend is that increasing the number of epochs results in better convergence. However, we can not simply keep on increasing epochs as not only it requires more computational power but it might lead to overfitting too. Overfitting in such cases can be handled by early stopping.

Hyperparameter values are fixed as given in the table below:

Hyperparameter	Value
n_dim	3
n_steps	50
lbeta	1e-5
ubeta	1.28e-2
scheduler_type	linear
batch_size	1024

Table 2: Hyperparameter values

Q 2. **Model complexity** - *How complex should our model be?* - Unlike number of training or diffusion steps, there isn't a concrete way to measure model complexity. A good proxy for measuring complexity could be simply the depth of the neural network you are using. For example: a single **torch.Linear** layer model can be considered very simple while a network with 10 **torch.Linear** layers can be considered very complex for our use case. Depending on your model architecture, you should define a complexity scale of models. Train at least 3 models with varying model complexity clearly stating your measure of model complexity and report your observations about sample quality similar to above question.

Answer: 3 models with varying complexities are trained. Their details are described below:

1. **Model 1:** One hidden layer of 64 neurons is used. ReLU layers are also used after each layer. Input and output layer contain 3 neurons each.

2. **Model 2:** Two hidden layers of 64 neurons each are used. ReLU layers are also used after each layer. Input and output layer contain 3 neurons each.

3. **Model 3:** Three hidden linear layers of 64, 128 and 128 neurons are used. ReLU layers are also used after each layer. Input and output layer contain 3 neurons each.

The results are reported below:

Model complexity	**EMD (sin_5_5)**	**EMD (helix)**
Model 1	124.39	120.17
Model 2	114.8	102.78
Model 3	**93.17**	**89.78**

Table 3: EMD values on test set corresponding to different model complexities for both datasets

Clearly on increasing model complexity, EMD decreases for both datasets. However, we can not simply keep on increasing complexity as it might lead to overfitting. Also, we tried using dropout but it led to higher loss as more information got lost.

Note: These results correspond to training on a small number of epochs (100). We had to restrict ourselves to 100 epochs due to time constraints. However, we expect EMD values to be even better if we train for more number of epochs.

Hyperparameter values are fixed as given in the table below:

Hyperparameter	**Value**
n_dim	3
n_steps	50
n_epochs	100
lbeta	1e-5
ubeta	1.28e-2
scheduler_type	linear
batch_size	1024

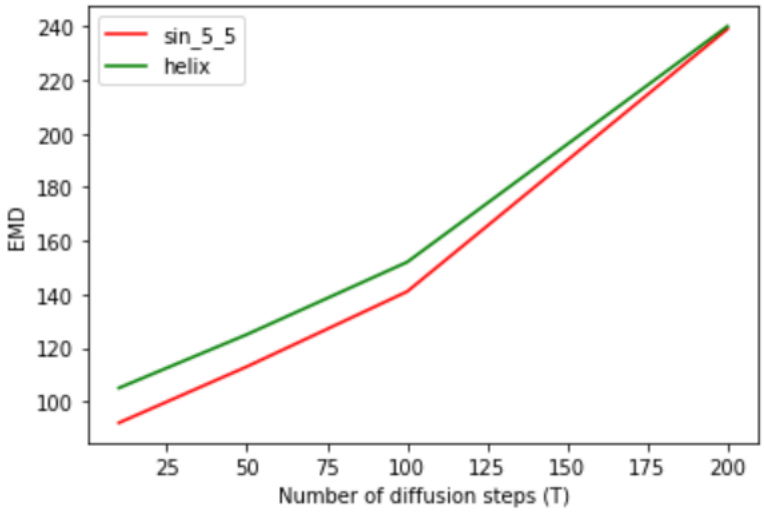
Table 4: Hyperparameter values

Q 3. **Number of diffusion steps (T)** - *Are the gains marginal when increasing T?* – Train at least 5 DDPMs for T = 10, 50, 100, 150, 200 respectively, keeping all other hyperparameters fixed. Similar to above 2 questions, report your observations about impact of diffusion steps on sample quality. You may also use the diffusion animation function we have provided to track the trajectories of few samples throughout the diffusion process and use the trajectory to support your claims.

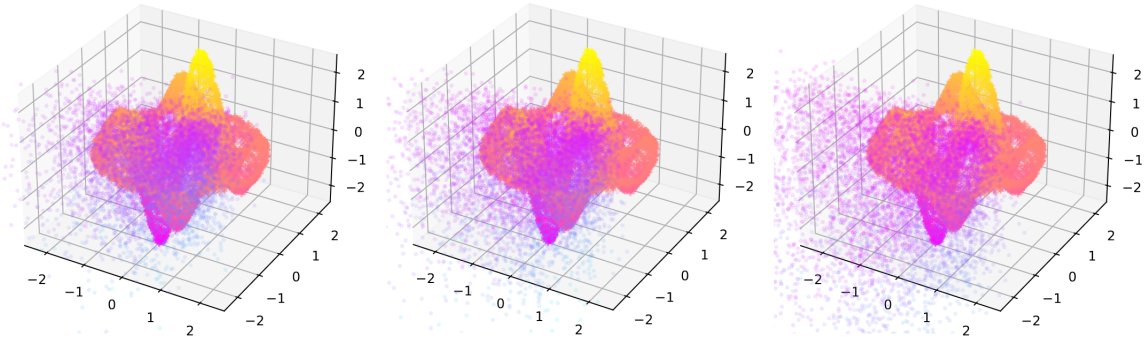
Answer: The results are reported below:

Number of diffusion steps (T)	EMD (sin_5_5)	EMD (helix)
10	**92.4**	**105.46**
50	113.59	125.74
100	141.34	152.31
150	173.58	180.76
200	239.67	240.56

Table 5: EMD values on test set corresponding to number of diffusion steps for both datasets



Clearly, the EMD value increases with increase in T. The explanation for this trend is that increasing T results in more destruction of structure in the input data. This claim is also supported by the following sample figures for the sine dataset corresponding to T = 50, 100, and 200 respectively.



Note: These results correspond to training on a small number of epochs (100). We had to restrict ourselves to 100 epochs due to time constraints. However, we expect EMD values to be even better if we train for more number of epochs.

Hyperparameter values are fixed as given in the table below:

Hyperparameter	**Value**
n_dim	3
n_epochs	100
lbeta	1e-5
ubeta	1.28e-2
scheduler_type	linear
batch_size	1024

Table 6: Hyperparameter values

Q 4. **Noise schedule** - *Which noise schedule works the best?* - The authors mostly use a linear noise schedule throughout the paper. This question invites you to explore other noise schedules. At the very least, study the effect of hyperparameters **lbeta** and **ubeta** on the quality of generated samples. Try at least 5-10 different noise schedules and report your observations. This is an open ended question, credit will be given to those who explore alternate noise schedules (such as cosine) and for motivating the need of the newer noise schedules.

Answer: The results are reported below:

Noise Schedule	**EMD (sin_5_5)**	**EMD (helix)**	**NLL (helix)**	**NLL (sin_5_5)**
Linear	**76.59**	**125.74**	3.2951	**2.8172**
Hyperbolic secant	95.76	130.69	3.4198	3.3724
Cosine	103.02	144.297	3.3975	3.3915
Scaled linear	123.02	134.276	**3.2714**	3.0915
Sigmoid	93.02	130.729	3.4032	3.4701

Table 7: EMD values on test set corresponding to different noise schedules for both datasets

We tried out 5 different noise schedules. Linear noise schedule provided the best results in all cases except on helix dataset compared by NLL metric.

Note: These results correspond to training on a small number of epochs (100). We had to restrict ourselves to 100 epochs due to time constraints. However, we expect EMD values to be even better if we train for more number of epochs.

Hyperparameter values are fixed as in previous questions. Now, we provide results obtained from our study on effect of hyperparameters lbeta and ubeta in the table below.

lbeta	**ubeta**	**EMD (sin_5_5)**
1.28e-2	1e-3	106
1.28e-2	1e-5	132
1.28e-1	1e-6	809

Table 8: Effect of lbeta and ubeta on quality of generated samples in terms of EMD

Clearly, as we widen the range of lbeta and ubeta, EMD starts exploding.

Q 5. **Bonus Task** - *FashionMNIST* - So far all our experiments have been based on a carefully curated toy dataset. However, the DDPM model is shown to be excellent at generating high quality image samples as well. In this part, we invite you to use all the insights you have gained during the assignment to train a DDPM on an image dataset called FashionMNIST. Notice that you will also have to implement dataloaders for this dataset yourself before proceeding. A straightforward approach to this could just be setting n_dims=784 and simply retraining your best diffusion model on FashionMNIST. Report whether this straightforward approach worked well or not along with your possible explanations. You are also welcome to change the model architecture completely to something that is more popular in vision tasks.

If you are able to get a DDPM that produces good samples, study if you can get a control over the generation. For example, if you know 2 vectors that take you to a half sleeved shirt and to a full sleeved shirt respectively after the reverse process, is there a combination of these vectors that take you to a three-quarters sleeved shirt? Figure 9 from the DDPM paper can be a motivating starting point.

Answer: We have tried out implementing DDPM on FashionMNIST dataset. First of all, we implemented data loaders in a Colab notebook and converted the FashionMNIST data corresponding to label 0 (T-shirt/top) into npy files. There were a total of 6000 instances in training set and 1000 in test set. All these instances were of 784 dimensions.

The figures obtained from the implemented DDPM are attached below. First figure corresponds to intermediate step of adding noise and second figure corresponds to final step of noise addition process.