Deep Learning HW3

Text Generation & Latent Space Visualization

This assignment explores two fascinating aspects of deep learning, using sequential models for text generation in natural language processing (NLP) and visualizing latent representations in generative models.
Through this assignment, you will gain hands-on experience with key concepts in sequence learning and a taste of generative AI, helping you build an intuitive understanding of model capabilities and representations.

Part 1: (Song Lyrics Generation)

The first task you are going to tackle is associated with text generation.

The first step is to download the data from the website.

The data is given as a csv file which contains the artist name, song name and song lyrics.

Study the data:

Import the file into your notebook and showcase a couple of rows from it.

Print all of the artist's names and show how many songs are associated with each artist.

Print the size of the dataset and how many songs are there.

Print the average number of characters and words of a song for all songs lyrics in the dataset (the average length of a song).

A nice to way to visualize text data with repeated words is by using a word cloud image.
Go over all the songs lyrics and print the top 3 most used words in the songs.

Then use a word cloud image to visualize the data, the most used words should appear the biggest in the word cloud.

Preprocessing:

The procedure for preprocessing data for NLP tasks is very straightforward.

Create a corpus from the given lyrics.

Remove any unrequired characters from the corpus, to do that you have to look in the corpus and check out if you got unnecessary symbols such as foreign language characters … etc.

Encode the data to make it ready for training the model.

Model Definition and training:

Use an LSTM model that can handle sequential data, as well as any loss function and optimizer of your choosing that you think is suitable for this task to train the model on the data.

Train the model on the lyrics such that it can be used to generate new lyrics afterwards, train it for 100 epochs and plot the training loss as a function to the number of epochs.

Evaluation:

Define a function to generate new lyrics such that it takes a starting string and a character count.

Def LyricsGenerator(starting_string, char_count)

The model should take the starting string as a prior and start generating new lyrics after that string, and it should stop generating once it reaches the character count.

In class, we learned multiple generation strategies (such as top-k, top-p …), make sure you use a suitable strategy.

Since you are going to experiment with multiple strategies, write in a text box the strategy you used and how the algorithm works in your case, as well as a couple of results from previous strategies you didn't end up choosing, showcase the difference between the generations.

To evaluate the model, we will generate 3 songs with different starting strings and a max character count to be the average character count of the songs in the dataset (which you extracted before).

Song 1 starts with " it's not a silly little moment …"

Song 2 starts with your favorite song's first sentence, write down which song you chose in a text box above the cell,
" — type your favorite song's starting string —"

Song 3 starts with an empty string "", this should generate a new song from scratch.

In the end you should have a minimum of 2 generations per song using different strategies.

Part 2: (Latent Space Visualization)

In deep learning, the latent space represents a compressed, abstract representation of input data learned by a model.

Exploring this latent space helps us understand how models learn and group similar data points.

In this assignment, you will train an autoencoder to project the MNIST images into a lower-dimensional latent space, and visualize the generations as a function of latent codes.

You will then explore how these representations capture the structure of the data, and visualize the linear interpolation between different digits.

By the end of this assignment, you will better understand how neural networks encode meaningful features of input data.

To start, download the given notebook from the website.

The preprocessing part and the training loop are already written.

your task is to implement:

1) The Encoder
2) The Decoder
3) The Variational AutoEncoder
4) The Following Loss Function from scratch:

$$\mathscr{L}(x,\hat{x},\mu,\sigma) = \underbrace{\text{BCE}(\hat{x},x)}_{\text{Reconstruction Loss}} + \underbrace{-\frac{1}{2}\sum\left(1+\log\sigma^2-\mu^2-\exp(\log\sigma^2)\right)}_{\text{KL Divergence}}$$

$$= -\sum\left[x\log(\hat{x})+(1-x)\log(1-\hat{x})\right]+\left(-\frac{1}{2}\sum\left[1+\log(\sigma^2)-\mu^2-\sigma^2\right]\right)$$

such that :

- x is the original value
- $\hat{x}$ is the reconstructed data from the model
- μ is the mean of the latent space distribution
- σ is the variance of the latent space distribution

you may **not** use pre-built loss functions from pytorch

5) A function to generate and plot a handwritten digit
6) A function to plot the latent space
7) A linear interpolation function between 2 randomly selected digits

Optional: The training loop is set for 50 epochs, you can modify it and add a piece of code that generates an image of a digit every 10 epochs, you should see how the model is learning and generates better images each time.

Running the code with your implementations should get you 2 latent space visualizations with different scalabilities.

In the end result you should see linear interpolations between different digits like we saw in class, and clearly visualize where each digit class is clustered in the space.

For the final part, you will implement a function which uses the latent codes of 2 randomly selected images from 2 different randomly selected digits from the dataset.

Use the trained VAE encoder that you implemented to obtain their latent codes.
Apply the following interpolation function:

$$\hat{x} = (1-t)x_1 + tx_2 \quad t \in [0,1]$$

s.t x_1 x_2 are the latent codes of the samples and $\hat{x}$ is the interpolated latent code.
Like we learned in class, t should increase in small steps.
Pass each interpolated vector through your implemented decoder in the VAE to reconstruct the corresponding images.

Plot the reconstructed images as a function to t, arrange them in a nice way to showcase the smooth transition between the sampled digits.

Bonus points will be given for a rendered video of the transmission.

Note: you may not change the given preprocessing code, but you may add extra arguments/functions to help do your task.

Submission:-

For this assignment, make sure your submission is clear and well-organized.
You should include a notebook for each part of the assignment with all code cells run and explained.
There's no need to put the files in a zip file, just submit your 2 ipynb files with their names being
HW3_PT1_ID1_ID2.ipynb & HW3_PT2_ID1_ID2.ipynb
Add text cells to introduce each section, explaining what you did and why. Keep your notebooks tidy and easy to follow, with the names of all group members at the top.

In each notebook, include comments in your code and use text to describe what you're doing in each section. Visualize your results with plots that help explain the data and model performance.

Finally, print each model you use, showing the layer details and the total number of parameters.

Only the pytorch library can be used as a deep learning tool to work with the models, you can't use keras or tensorflow under any circumstances, you may use other libraries for manipulating the dataset and preprocessing as long as it doesn’t affect the model definition within pytorch.