Challenge Problem 3 Reflection

Bobby Albani

Overview

This challenge problem offered a number of different paths in creating a solution. Beyond the work we did for demo 9 using diffusion models to inpaint the masked bike images, the challenge problem also offered us text and parametric data to hone our model results. This was crucial as, unlike the demo, the challenge problem did not offer any target data for our model to train towards. The addition of this challenge made the problem much more interesting and, because of this, I decided to explore a number of different possible solutions before reaching my final solution.

Diffusion Model

Looking at the VAE solution that was initially given to us in the starter code for this challenge problem, my immediate instinct was to implement a diffusion model with some guidance function. Thankfully, having been given a pre-trained diffusion model for this purpose, I was able to quickly implement this idea as part of my solution. This change gave me an output value of around 0.96, slightly above, but not meaningfully larger than the output from the VAE. After implementing this, I began to think of ways to improve my output, specifically ways to incorporate the parametric and text data that was given to us for this problem. Having no real bearing on what the output should look like, the additional data is vital in creating a more convincing output.

Parametric Data

The first piece of data that I decided to look at was the parametric data. As stated in the challenge problem starter code, the first hurdle to use the parametric data was that it was riddled with NAN feature values, or features that were undefined for each sample. In order to overcome this problem, a solution that was suggested in the challenge problem overview video was to use Imputation. This would allow us to fill in the gaps of the data by taking the averages of known values of the same feature in different samples. I was able to do this by implementing the SimpleImputer method provided by sklearn. After filling in the missing datapoints, the next challenge I faced was how to implement it into my solution.

I spent much time looking at the data to understand what I was looking at. However, simply observing the data points using different plots did not help me to understand what each parameter ment. I began to believe that a good way to implement this into a model would be to create a hybrid neural network containing both a U-net and a DNN. This would allow the model to learn from both the image data and parametric data continuously and learn better relationships between the two data types. Although this seemed to be the best solution in theory, in practice, I found many issues with combining the two network types and ultimately could not find a working solution on training the data, not to mention the time constraints that I had that prevented me from training the model.

Text Data

Next, I decided to look into the textual data. The first challenge I saw with implementing this was how to represent the data in a meaningful way for the model to understand. I looked into a number of different natural language processing techniques before looking into the methods used in demo 10. By using multimodal text imbeddings with CLIP, I realized that I could compare image and text data. This could be used to calculate the loss between generated images and intended word descriptions of the image. When I realized that the model could be improved by changing the loss function, I turned my head back to the diffusion model guidance function.

Instead of only calculating the loss between the generated image and the non-masked portions of the image, I decided to also calculate the loss between the generated image and the text description that came with the original masked image. By doing this, we could combine the benefits of the non-masked image data and the text data to create a better image. I did this by defining a new text loss function that found the cosine similarity between the generated images and text descriptions. This loss was then added to the original image loss to create an all-encompassing error calculation. It is this solution that I was able to ultimately add to my diffusion model and produce effective results.

Conclusion:

This challenge problem encouraged me to think deeply about different generative solutions to a design problem that we have been talking about for a while in this class. This being the first time I explored different generative models apart from OpenAI or Dall-E, it helped me to understand the capabilities of the generative models that we have on hand today and the different ways in which they can be used to solve problems. Just as we did for this challenge problem, I see now that these generative models can be used for many different design applications to generate more efficient solutions to everyday tools and machinery.

What I found most difficult about this challenge problem was the time management and debugging. Due to long runtimes for testing, especially for model training and application, I spent a lot of time right after the assignment was released exploring different solution paths. However, due to bugs and other hurdles, I was unable to find a working method until closer to the deadline. This made it much harder for me to iterate through my code to see what worked and what didn't. Despite these challenges, I believe that I was able to come to a well formulated solution, although I was unable to improve it to make better scores.

The first expansion to my solution that I would've liked to have made was increase the number of data points. As I was more focused on the different data points I could use, I lost sight of the possible strides I could've made by simply increasing the size of my dataset using the masking function that was given. Another expansion to my solution that I would have liked to have added was, as explained above, the incorporation of the parametric data. By making these changes, I believe that I would have been able to generate much better results.