CENG 796 - Peer-review form

Reviewed project ID: Group 01

Reviewed project's title (title of the paper): DiffDis: Empowering Generative Diffusion Model with Cross-Modal Discrimination Capability

Reviewer name(s): Enes Şanlı, Hamza Etcibaşı

Instructions:

- Answer = *Yes*, *No* or *Partial*.
- You may expand sections as necessary.
- For most questions, you do not need to add comments, unless the instructions tell you otherwise.
- "Notebook" refers to "Jupyter Notebook" file that is expected to be named as main.ipynb

Question	Answer	Comments
Contains a jupyter notebook file	Yes	
Notebook is located at <project_root>/main.ipynb	Yes	
Notebook's first section contains paper information (paper title, paper authors, and project group members' name & contact information) Some good examples: see group03, group10, group11 (and a couple of other groups).	Partial	The authors of the paper are not specified
Notebook contains a section for hyper-parameters of the model.	Yes	
Notebook contains a section for training & saving the model.	Yes	
Notebook contains a section (or a few sections) for loading a pre-trained model & computing qualitative samples/outputs.	Yes	
Notebook contains reproduced plots and/or tables, as declared.	No	
Notebook contains pre-computed outputs.	Yes	

Data is included and/or a proper download script is provided.	Yes	
Notebook contains a section describing the difficulties encountered.	Yes	The group had some issues reported in the challenges section, so after reading the paper, we shared our understandings in the additional comments section below.
The paper has achieved its goals and/or explained what is missing.	Partial	
The notebook contains a section that reproduces the figure(s) and table(s) declared in the goals.	No	They have mentioned that they haven't achieved satisfactory results yet, mainly because the training of the model is quite challenging. Consequently, they haven't been able to reproduce the figures and tables accurately.
The notebook also reports the original values of the targeted quantitative results, for comparison.	No	Since the group has used a dummy dataset, they could only show a random generated output image. Hence, observed neither quantitative results nor meaningful qualitative results.
MIT License is included.	Yes	
As the reviewer(s), you have read the paper & understood it.	Yes	
Implementation of the model seems correct.		According to the `encoder.py` and `clip.py` files, the encoder components utilize CLIP and Variational Auto Encoders (VAEs). For text encoder they use the CLIP model, for image encoder they use VAE model. For text-conditioned image generation, they employ a U-Net based latent diffusion model according to diffusion.py, similar to what's described in the paper. They also implement attention blocks for UNet in attention.py file. The encoder parts and image generation components appear to be correctly implemented. However, it seems that they have not yet implemented transformer models for the text-image alignment part.
Notebook looks professional (in terms of notation, readability, etc.)	Yes	They divided it into sections according to topics, but the functions of the sections are generally stated in the comments. However, titles can be given to sections using markdowns.
Source code looks professional (in terms of coding style, comments, etc.)	Yes	Yes, the source code looks professional. It follows a clear coding style of Python, includes informative comments, and utilizes appropriate naming and formatting conventions.

Additional comments:

1) The paper did not provide explicit information regarding whether the transformers employed for the image-to-text alignment task were part of a separate architecture or integrated within the proposed UNet middle blocks. Given this lack of clarity, our implementation treats the transformers as a distinct architecture.

Answer 1: According to the paper, the transformer employed for the image-to-text alignment task is integrated within the proposed UNet middle blocks; it is not a separate architecture. This integration is mentioned in the section describing the architecture of the model, where it states:

"Besides, the Φθ (the decoder part of the UNet) also contains a unique transformer with M transformer block and a linear predictor. The unique transformer follows the middle block of the UNet to obtain more semantic information."

Although the authors use pre-trained autoencoder and UNet from Stable Diffusion v1-1,, the transformer part should be trained from scratch. In the provided code, the group implemented this part as UNET_AttentionBlock separately. The correct implementation should be taking the features from the middle block of UNet and process in Attention block starting from scratch with 6 transformer blocks with 6 transformer blocks with 768 model width and 64-dim attention heads (we observed that the implementation is different in the provided code.)

2) The paper did not specify which blocks were to be modified to incorporate the dual stream deep fusion blocks. In our implementation, we have chosen to integrate these blocks into both the downsample and middle blocks of the original UNet, also known as Stable Diffusion. Notably, we did not apply these changes to the upsample layers, as they are not utilized in the image-to-text alignment task.

Answer 2: According to the paper:

“The transformer’s input is the concatenation of the flattened image’s feature map and the text query outputted from the middle block of UNet.”

Therefore, the integration process should focus solely on the middle blocks, ensuring that the dual-stream deep fusion attention mechanism is effectively incorporated. Additionally, it's crucial to concatenate the flattened image's feature map with the text query from the middle block and feed them into the unique transformer contained in Φθ.

It's worth noting that there's no need to modify the upsample layers, as they are not utilized in the image-to-text alignment task, as mentioned in the paper.

3) The term 'text query' was used in the paper without a clear definition. In our interpretation, we have chosen to represent the text query as the normalized average of the output from the text encoder.

Answer 3: The distinction between "text condition" and "text query" in the paper is important for understanding how textual information is utilized in the model we think. So

Text Condition (c): This represents the token-wise representation of the text prompt. In essence, it encapsulates detailed information about the text, preserving the individual tokens and their respective embeddings. The text condition is used for conditioning the generation process in the paper, providing fine-grained control over the output.

Text Query (e): On the other hand, ***the text query is the normalized global representation of the text prompt*** *(so the implementation seems correct according to us).* Instead of preserving token-level information, it aggregates the overall meaning or context of the text. This global representation serves as a high-level summary of the text content, which can be useful for the alignment task.

The reason for using both the text condition and text query simultaneously we guess is to leverage the strengths of both representations. The text condition offers detailed information that can guide the generation process at a fine-grained level, while the text query provides a broader context or summary of the text's meaning. By incorporating both, the model can achieve a more comprehensive understanding of the textual input and produce more accurate and contextually relevant outputs.

4) The paper did not provide a clear methodology for the concatenation of the hidden latent image and the output of the fully connected layer. In our implementation, we expanded the output of the fully connected layer from a shape of (Batch Size, Channels) to (Batch Size, Channels, Height, Width), enabling its concatenation with the latent image, which also has a shape of (Batch Size, Channels, Height, Width).

Answer 4: The paper indeed lacks explicit details on the concatenation process, but it does specify that the input to the transformer should be the concatenation of the flattened image's feature map and the text query outputted from the middle block of the UNet. Based on this, we recommend the following:

Flattened Image's Feature Maps: The latent image's feature map should be flattened before concatenation. which in this case would collapse the spatial dimensions (Height x Width) into one dimension while preserving the channel dimension.

Fully Connected Layer Output: Keep the output as (Batch Size, Channels) without expanding it to include spatial dimensions as this could introduce unnecessary complexity.

So that you can concatenate these tensors along the channel dimension, resulting in a concatenated tensor with shape (Batch Size, Combined Channels).

*5) The paper did not provide explicit instructions on how the fully connected layer projects the text query back into the text embedding space. We assumed that it generates an output with a shape of (Batch Size, Width * Height, 768). We then computed the normalized average of this output along dimension 1, resulting in an output of shape (Batch Size, 768). This output serves as the hidden text query that is input to the next layer.*

Answer 5: According to paper text query dimension $e \in R^{1 \times dy}$ so we think when concatenating with hidden latent image, the size will be image = (Batch Size, Width * Height, 768), text query =(Batch Size, 1, 768), concatenated result is =(Batch Size, Width * Height + 1, 768). Then, after the self attention layer they should split the embedding into 2 embedding, one is hidden latent image with attentioned the other is text query with attantioned. Finally, when they want to give embedding to Fully Connected layer, they just give text query=(Batch Size, 1, 768)