NTU AI 2024 HW1 Report

B10902037 Yu Xiang Luo

Task 1

1. Briefly describe how you implement the two models:

   In this assignment, I utilize the huggingface packages to conduct the experiments. Below is the main package I use:

   - `transformer.AutoModel...` → Load models.
   - `datasets.load_dataset` → Load datasets.
   - `torch.utils.data.DataLoader` → Process data in batch to inference faster.
   - `evaluate` → Provide functions for Corpus BLEU, ROUGE, and METEOR.

   I also converted the captions to lowercase and removed punctuation for better evaluations.

2. Metrics Result:

	MSCOCO-Test				flickr30k			
	BLEU	ROUGE-1	ROUGE-2	METEOR	BLEU	ROUGE-1	ROUGE-2	METEOR
BLIP	0.3237	0.4023	0.1594	0.2776	0.2398	0.3206	0.1091	0.2033
Phi-4	0.2560	0.3971	0.1517	0.3325	0.2658	0.3636	0.1400	0.3043

3. Analysis:

   - In the MSCOCO experiments, BLIP outperforms Phi-4 on BLEU metrics but is outperformed by Phi-4 on METEOR. This suggests that BLIP's output aligns more closely with the specific wording used by MSCOCO annotators, leading to higher BLEU scores. In contrast, Phi-4, as a larger model, likely generates captions with a more diverse vocabulary, improving its METEOR score by capturing broader semantic similarities.
   - In the Flickr30k experiments, Phi-4 outperforms BLIP across all metrics, indicating that Phi-4 is more effective for this dataset.
   - Comparing the two datasets, we observe that the overall metric scores are higher for MSCOCO, suggesting differences in dataset complexity, with MSCOCO potentially being easier for models to perform well on.

4. Case Study:

   In Figure 1-a, the BLIP model generated the caption: *"a man in a white shirt,"* which fails to mention that there are actually five men in the image. In contrast, the Phi-4 model produced the caption: *"Men in business attire are standing around a van,"* which captures more detail and provides a more accurate description. However, when comparing these outputs to the reference captions, BLIP surprisingly achieved higher ROUGE-2 and METEOR scores than Phi-4. This example highlights a key limitation of rule-based language metrics: they tend to prioritize surface-level similarity and rigid sentence structures over semantic richness. In contrast, evaluating caption quality using cosine similarity between text embeddings could offer a more robust and meaningful measure of semantic alignment.

Figure 1-a: flickr30k image

Figure 1-b: Peanuts style by stable diffusion v3

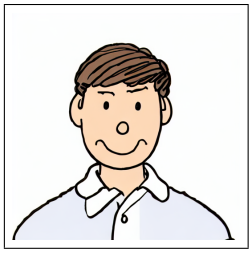

Figure 1-c: Peanuts style by stable diffusion v1.5

Task 2-1

1. Briefly describe how you implement task 2-1:
   - Generated captions with Phi-4 beforehand and stored in a json file.
   - Used `diffusers.StableDiffusion3Pipeline`.
2. Visualization:
   (a) Figure 1-b shows an example output.
   (b) Below are examples of success and failure. In the success samples, the images are indeed in the Peanuts style, and the hair color, hairstyle, clothes, and accessories are correctly generated. On the other hand, some failure samples show no Peanuts style at all.

Figure 2: Task 2-1: Success samples

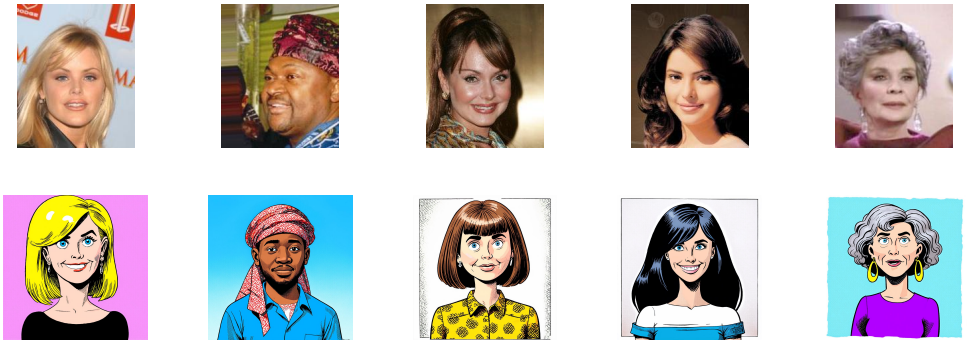

Figure 3: Task 2-1: Failure samples

(c) I tried to generate detailed captions so that the text-to-image (T2I) model would receive as many instructions as possible. I achieved this by asking the model not to describe the background but to focus on the avatar. This did not lead to a significant improvement. Then I tried to improve performance by clarifying the prompt to indicate a "Peanuts comic" style (I was worried the model might interpret "peanut" as "groundnut"). It did not work better either. In conclusion, I think zero-shot usage of Stable Diffusion might not yield a significant improvement.

Task 2-2

1. Briefly describe how you implement task 2-2:
   - Generated the captions by Phi-4 beforehand and stored in a json file.
   - Padded or resized image to 512 × 512.
   - Used `diffusers.StableDiffusionImg2ImgPipeline`.
2. Visualization:
   (a) You can see an example in Figure 1-c.
   (b) Below are examples of success and failure. In the success samples, the face orientation, clothes, and hair color are correctly displayed—although the fourth sample generated the wrong gender, it still captured many relevant features. In the failure samples, most of them fail to generate a proper face.

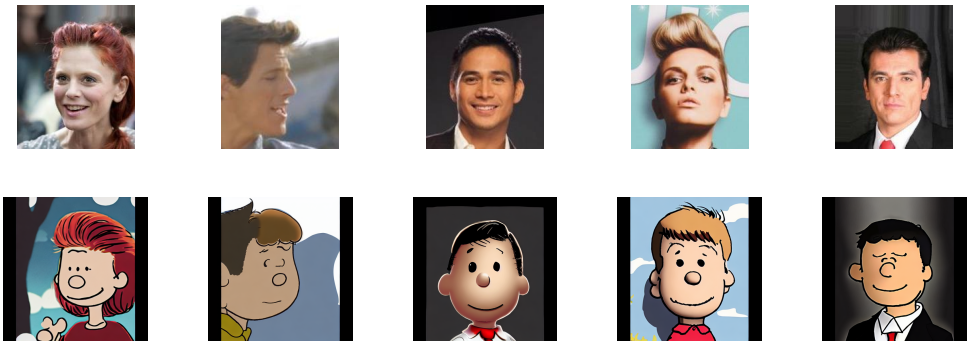

Figure 4: Task 2-1: Success samples

Figure 5: Task 2-1: Failure samples

   (c) In `diffusers.StableDiffusionImg2ImgPipeline`, there are two tunable parameters: `strength` and `guidance_scale`. I experimented with several combinations. With lower **`strength`** and **`guidance_scale`**, the generated image retains more features from the original content image but shows less of the Peanuts style. This trade-off is something we must consider.