Improving Text to Image Models using Semantic Understanding

Team Sarovar

August 26, 2024

1 Introduction

The major limitations of text-to-image generators are semantic understanding and common-sense reasoning. That is, input prompts that are too simple and not concise lead to low-quality image generation. We attempt to match this semantic representation of these simple narrative prompts to complex prompts via the use of LLMs, facilitating the SUR adapter to acquire powerful semantic understanding and reasoning capabilities.

Although CLIP is a powerful model for capturing the underlying image and text relations, it requires complex and detailed prompting for good results. This is because text-image tasks not only require understanding the semantics of the text but also figuring out the implicit information and knowledge grounded in the text. The CLIP encoder fails in all three common categories of text prompts in multi-modal visual question answering: "counting," "color," and "action." Our obtained results, as shown below, also confirm the claim made.

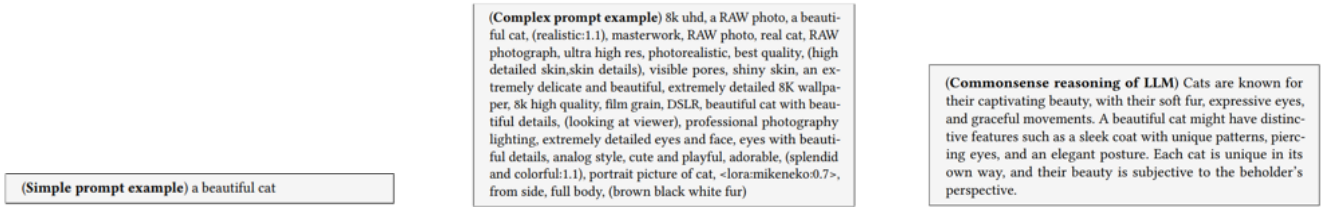
(a) A collection of seven vintage glass bottles in different shapes and sizes, arranged on a windowsill

(b) Five dogs

Figure 1: Results Obtained from Realistic_Vision_V2.0

2 Dataset

Each sample of the Dataset consists of a simple narrative prompt, its corresponding complex prompt and high quality image.



(a) Simple Prompt (b) Complex Prompt (c) Inference by LLM

Figure 2: Glimpse of the Dataset

3 Reference Implementations

- SUR Adapter Github Repo
- Ella GitHub Repo

3.1 SUR Model Architecture

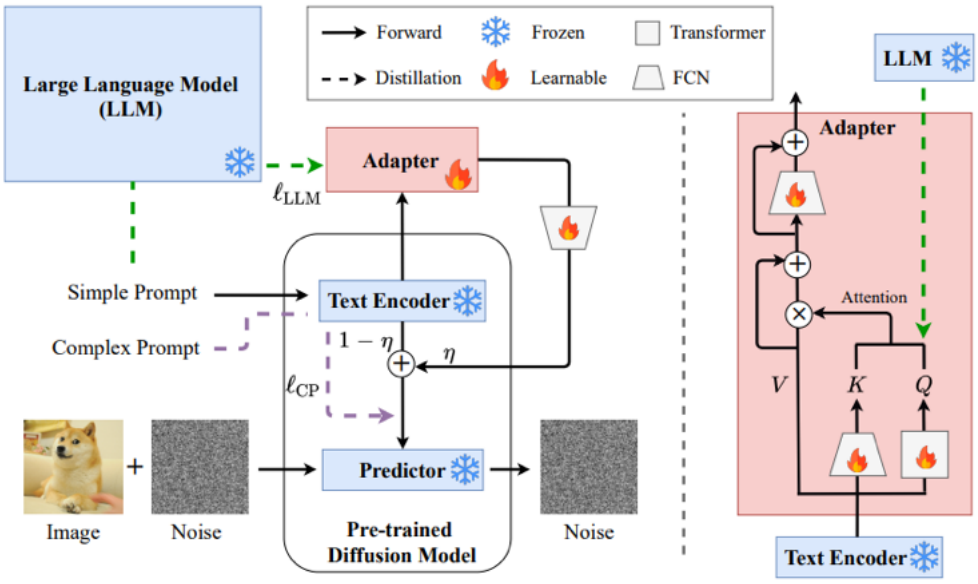


Figure 3: Model Architecture

The Architecture consists of two trainable neural networks g_{Ada}, g and with parameters ϕ_1 and ϕ_2

1. Adapter
   - **Adapter**:
   $$g_{\text{Ada}}(f_{\text{En}}(p_{is}); \varphi_2) = V_i' + V_i + h_1[V_i' + V_i], \tag{1}$$
   where $V_i = f_{\text{En}}(p_s^i)$ and $V_i' = V_i \otimes \text{att}_i$.
   - The output of the adapter is transformed using $g(\cdot; \varphi_1)$, therefore:
   $$g(g_{\text{Ada}}(f_{\text{En}}(p_{is}); \varphi_2); \varphi_1) = g(V_i' + V_i + h_1[V_i' + V_i]; \varphi_1), \tag{2}$$
   - Semantic information input to the predictor:
   $$c'_{\text{LLM}} = \eta\ c_{\text{LLM}} + (1-\eta) \cdot f_{\text{En}}(p_s^i). \tag{3}$$
2. - Final Loss function is given by:
   $$l_{\text{total}}(\phi) = \lambda_1 \cdot l_{\text{LLM}}(\phi) + \lambda_2 \cdot l_{\text{CP}}(\phi) + l_t^{\text{simple}}(\phi)$$

- The intermediate loss expressions areas follows:

$$\ell_{\text{CP}}(\phi) = \text{KL}\left(\frac{c'_{\text{LLM}}}{\tau}, \frac{f_{\text{En}}(p_c^i)}{\tau}\right)$$

$$\ell_{\text{LLM}}(\phi) = \text{KL}\left[\frac{W_0 f_{\text{LLM}}(p_{is})}{\tau}, \frac{Q_i}{\tau}\right]$$

3.2 Plausible Explorations

1. **Adaptive Improvement to Prompt:** One important point to note is that η doesn't vary based on the input. Therefore, irrespective of the comprehensiveness of the input, a fixed convex combination of f_{En} and $\eta \times c_{\text{LLM}}$ is always used to aid our prompt. This may be an inhibiting factor if our prompts are well-detailed or the Adapter network is not very well trained. We plan to tackle this by making the factor η learnable, adding an additional fully connected network (FCN) which takes as input the text embedding. This allows the architecture to adapt to the text prompt.
2. **Retrieval-Augmented Generation:** Using RAG we plan, to provide additional information to the LLM base prompts while performing knowledge distillation, prompts like draw image of Narendra Modi, will get translated to "Gujarati Guy, White Beard, White Complexion, Old Person, Specs".
3. **Multi-LLMs Knowledge Distillation** There is a KL Distillation loss term involved in the final loss expression. A possible improvement could be to use multiple LLMs to further enhance the text encoder by introducing a KL distillation loss term for each LLM and appropriately weighing them based on the superiority of each LLM.

Note: During learning we freeze all the LLM parameters, the text encoder, the predictor in the pre-trained diffusion model.

4 Experimental Results

The following SUR Adapter results are shown on pre-trained weights published by the authors on $\eta = 0.01$. Our obtained results with increasing training steps are as shown below. For the code implementation, refer to the collab notebook available

4.1 Action Category

(a) Stable Diffusion Model

(b) SUR Adapter

Figure 4: Action Category: A chef tossing a pizza dough in the air in a kitchen

4.2 Colour Category

(a) Stable Diffusion Model

(b) SUR Adapter

Figure 5: Color Category: A couple wearing blue and yellow solid color clothes

4.3 Counting Category

4.4 Simple Prompts

We also conducted a few experiments on simple prompts such as "beautiful cat" to test whether there is any improvement in the semantic understanding of the SUR model compared to that of stable diffusion models with simple and complex prompts. However, we were unable to observe any significant change and have displayed the obtained results below:

(a) Stable Diffusion Model

(b) SUR Adapter

(c) Stable Diffusion Model using complex prompt

Figure 7: Simple Prompts for testing improvement in semantic understanding

4.5 Results Obtained Training

We trained the SUR Adapter with a default value of $\eta = 0.1$, but the results obtained were completely bogus even after 5000 training steps. Our obtained results with increasing training steps are as shown below. For the code implementation, refer to the collab notebook available here.

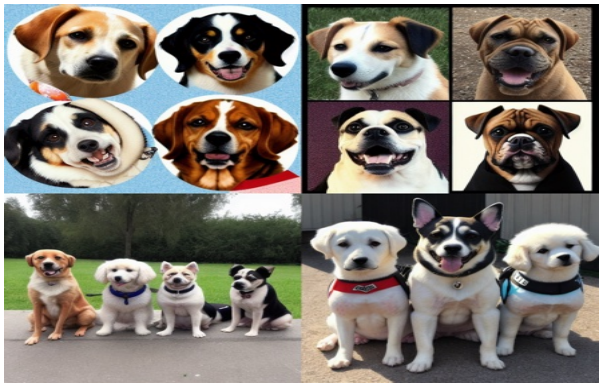
(a) “Four dogs” by Stable Diffusion Model

(b) “Four dogs” by SUR Adapter

(c) “Five dogs” by Stable Diffusion Model

(d) “Five dogs” by SUR Adapter

(e) “Six dogs” by Stable Diffusion Model



(f) “Six dogs” by SUR Adapter

(g) Stable Diffusion Model results for seven vintage bottles

(h) SUR Adapter results for seven vintage bottles

Figure 6: Counting Category: (a) “Four dogs” (b) “Five dogs” (c) “Six dogs” (d) A collection of seven vintage glass bottles in different shapes and sizes, arranged on a windowsill

(a) 1000 training steps (b) 2000 training steps (c) 5000 training steps

Figure 8: Generated images with varying training steps for the prompt: "An aristocratic maiden in medieval attire with a headdress of brilliant feathers"

5 Proposed Project

5.1 Understanding the Drawbacks of SUR

The architecture proposed in the SUR paper demonstrates poor practical performance, as none of the paper's claims were substantiated in either the pretrained weights provided by the authors or in our own model trained with significant iterations. Our investigation revealed several key factors contributing to the inefficiency of the aforementioned architecture:

- The custom SUR dataset developed by the authors lacks adequate quality; many instances lack a discernible relationship between the simple prompt and the complex prompt, resulting in subpar image generation. This highlights unrealistic expectations for SUR to comprehend and execute such tasks effectively.
- The training methodology employed for SUR, particularly the integration of LLM knowledge distillation, appears ineffective in imbuing the extensive knowledge of LLMs into the SUR adaptor. Only the hidden state output of the LLM is encoded into a format compatible with CLIP via an untrained linear transformation, without a provided rationale or plausible explanation for why a random matrix multiplied by the LLM hidden state would accurately capture LLM information.
- The utilization of KL loss for distillation implementation seems intuitively incorrect, a sentiment further validated by our experimental results.
- The suggested low value of η, the influence parameter of SUR by the authors, diminishes the impact of SUR on the conditioning of the U-Net, thereby hindering its effectiveness.

5.2 Proposed Changes

Now to fix the SUR adapter and generate results that are an improvement over the existing State of the Art(SoTA) text to image diffusion model `runwayml/stable-diffusion-v1-5`, we propose the following modifications: Now, to enhance the performance of the SUR adapter and produce results surpassing the existing State of the Art (SoTA) text-to-image diffusion model `runwayml/stable-diffusion-v1-5`, we propose the following modifications:

- **Dataset**: Although the task may appear straightforward, it proves to be rather challenging. SoTA text-to-image models are trained on diverse and extensive datasets, enabling them to generalize well across various domains. This implies that employing simplistic `(Text, Image)` pairs does not train the SUR adapter, mainly results in SUR adapter learning identity transform. Instead, we advocate for a dataset where the prompts are exceptionally simple yet attempt to convey complex images. Our exploration led us to the `pokemon-llava-captions` dataset, depicted in Figure 9. Notably, this dataset features images with diverse themes and simple prompts, requiring the SUR adapter to infer complex attributes from straightforward prompts and adapt to various thematic contexts.
- **Knowledge Distillation**: The previous knowledge distillation process was ineffective due to the flawed construction of the loss function. In our proposed revised methodology, we suggest a modified approach. Firstly, instead of solely utilizing the hidden state of the LLM, we advocate for extracting the entire inference from the LLM. Additionally, we employ prompt engineering techniques to enhance the quality of the output. This approach not only ensures the presence of LLM knowledge verbatim but also facilitates subsequent

stages. Encoding the LLM's inference using the CLIP encoder yields the complex prompt embedding, aligning both in the same domain, which is crucial for designing an effective loss function.

- **Loss Function**: Employing KL divergence on the embeddings of the LLM linearly transformed into the CLIP encoder's output domain has proven ineffective for model evaluation. Consequently, with the aforementioned modifications, both the complex and simple prompts are encoded into the same space. Hence, constructing cosine similarity-based losses provides a more robust methodology for model evaluation and training.
- **Adaptive** η: To address this, we propose a straightforward solution. Initially, during testing, we employ a large value to allow SUR to influence the conditioning of the U-Net. Subsequently, we propose a modification that enables adaptive influence, adjusting based on the prompt complexity. For instance, simple prompts would entail higher influence, whereas detailed prompts would entail lesser influence, providing users with greater control.

Figure 9: Sampled instances from the `pokemon-llava-captions` dataset.

6 Dataset

As described before, the conditional U-Net is difficult to train, and therefore a lightweight solution is to enhance the conditioning of the U-Net, through the text embedding, this as experimented, not only improves the output image but also helps it adapt to images of different domains, like in our case where it adapts to the Pokémon theme! Essentially, when we trained on the llava dataset, we were learning the anime themes and terminology, and by using simple prompts we were able to generate good quality results.

7 Loss Function

Table 1: Comparison of Cosine Loss and Distillation Loss

Text 1	Text 2	Cosine Loss	Distillation Loss
Hello! How are you?	Hi, How your doing	0.144	147.963
Hello! How are you?	Bye! Mom where are you going	0.501	183.568
Hello! How are you?	White tiger and blue elephant	0.644	83.317
Hello! How are you?	Horse on the moon	0.603	68.026
Hello! How are you?	Donkey in Mars	0.608	72.141
Hello! How are you?	Violet lion and green donkey	0.646	81.022
Hello! How are you?	Purple cheetah and yellow hippo	0.658	87.245
Hi, How your doing	Bye! Mom where are you going	0.518	41.167
Hi, How your doing	White tiger and blue elephant	0.666	145.305
Hi, How your doing	Horse on the moon	0.617	123.277
Hi, How your doing	Donkey in Mars	0.625	115.777
Hi, How your doing	Violet lion and green donkey	0.684	151.838
Hi, How your doing	Purple cheetah and yellow hippo	0.668	145.109
Bye! Mom where are you going	White tiger and blue elephant	0.780	179.967
Bye! Mom where are you going	Horse on the moon	0.700	167.846
Bye! Mom where are you going	Donkey in Mars	0.709	145.205
Bye! Mom where are you going	Violet lion and green donkey	0.783	169.383
Bye! Mom where are you going	Purple cheetah and yellow hippo	0.786	153.516
White tiger and blue elephant	Horse on the moon	0.620	48.909
White tiger and blue elephant	Donkey in Mars	0.722	58.681
White tiger and blue elephant	Violet lion and green donkey	0.487	37.402
White tiger and blue elephant	Purple cheetah and yellow hippo	0.539	45.276
Horse on the moon	Donkey in Mars	0.306	22.842
Horse on the moon	Violet lion and green donkey	0.591	48.885
Horse on the moon	Purple cheetah and yellow hippo	0.703	58.616
Donkey in Mars	Violet lion and green donkey	0.560	45.688
Donkey in Mars	Purple cheetah and yellow hippo	0.745	58.434
Violet lion and green donkey	Purple cheetah and yellow hippo	0.471	34.948

The loss function suggested in the paper was KL divergence loss to align the semantic representation of simple prompts to the complex prompts and transfer knowledge of large language models (LLMs) to the SUR-adapter via knowledge distillation. As we can observe from the table, KL divergence loss isn't quite effective in capturing the similarity between text embeddings and, in fact, to some extent quite random. On the other hand, we observe that the Cosine loss captures the similarity in the semantic meaning of the prompts (e.g., 1^{st} entry in the table). We even confirm this by utilizing Cosine Similarity loss for aligning the embeddings.

7.1 Cosine Loss Function

One rationale for implementing this loss function is to compute the average along each dimension of the word embedding. This process can be likened to calculating the mean word[1], subsequently normalizing and comparing them, and penalizing the differences, thereby assessing dissimilarity.

```
def Cos_loss(llama_embeddings, llama_embeddings_1):
    clip_text_embedding1_avg = torch.mean(llama_embeddings, dim=1)
    clip_text_embedding2_avg = torch.mean(llama_embeddings_1, dim=1)


    # Normalize the embeddings
    clip_text_embedding1_normalized = torch.nn.functional.normalize(clip_text_embedding1_avg, p=2, dim=1)
    clip_text_embedding2_normalized = torch.nn.functional.normalize(clip_text_embedding2_avg, p=2, dim=1)

    # Calculate cosine similarity loss
    similarity_loss = 1-cosine_similarity(clip_text_embedding1_avg, clip_text_embedding2_avg)
    return similarity_loss
```

[1] Vector in the embedding space representing the sentence's meaning

8 Final Results for Our Modified SUR Adapter

We observe that the images are more accurate for the case of the SUR adapter and the images become more cartoonish as we increase the number of training steps.

8.1 Color Analysis

(a) Clip Stable Diffusion Model

(b) SUR Adapter 3000 steps

(c) SUR Adapter 4000 steps

(d) SUR Adapter 5000 steps

Figure 10: Analysis of results for the prompt: "A vibrant butterfly with iridescent wings in shades of blue, green, and purple, perched on a bright pink flower"

(a) Best Picture For Clip Stable Diffusion Model

(b) SUR Adapter 4000 steps

Figure 11: Analysis of results for the prompt: "The blue glass containing red juice"

8.2 Action Category Analysis

(a) Best Picture For Clip Stable Diffusion Model

(b) SUR Adapter 3000 steps

(c) SUR Adapter 4000 steps

Figure 12: Analysis of results for the prompt: "A chef tossing a pizza dough in the air in a kitchen"

(a) Best Picture For Clip Stable Diffusion Model

(b) SUR Adapter 4000 steps

Figure 13: Analysis of results for the prompt: "Giraffes eating trees"

Here once again, our modified SUR adapter's performance is comparable to the best performance of the Clip diffusion model with respect to the random seed. Here we once again observe that

8.3 Comparison using Complex Prompts With Best Clip Results

(a) "Three fluffy white kittens playing with a ball of yarn on a bright green carpet"

(b) SUR Adapter Results

(c) "A collection of seven vintage glass bottles in different shapes and sizes, arranged on a windowsill"

(d) SUR Adapter Results

(e) "An aristocratic maiden in medieval attire with a headdress of brilliant feathers"

(f) SUR Adapter Results

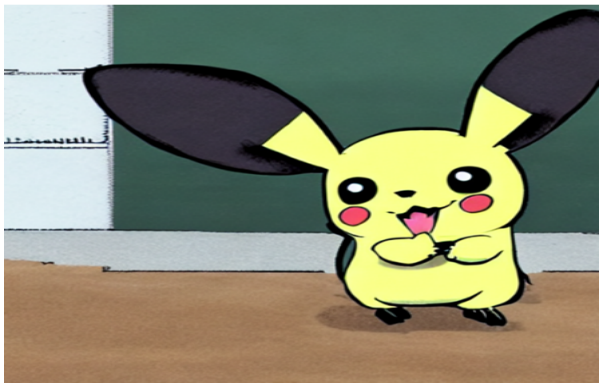
Figure 14: Analysis of Complex Prompts

Here we observe for (b) that the ball of yarn is also white and in fact the same color as that of the cat. This is a clear example for *attribute leaking*, where attributes specified in the prompt are correctly bound but some other elements in the scene are also wrongly bound with this attribute.

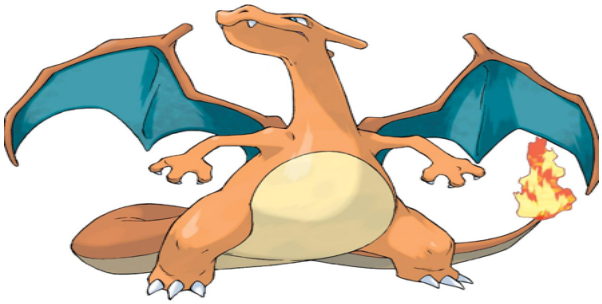
8.4 Pokemon Comparison for Fun!

We conducted an evaluation of the model using a selection of sample Pokémon prompts. For instance, prompts included descriptions such as "a yellow rabbit with red cheeks, long ears, and zig-zag tails." Here are the results obtained for various Pokémon prompts.

(a) Pikachu



(b) Pikachu by SUR



(c) Charizard

(d) Charizard by SUR Adapter

(e) Greninja

(f) Greninja by SUR Adapter

Figure 15: Analysis on Pokemon prompts

9 Analysis of New Loss

(a) "A golden sun setting over a calm ocean, with orange and pink hues appearing in the sky"

(b) "A gymnast performing a balance beam routine with graceful flips and twists"

(c) "A skateboarder doing a kickflip over a set of stairs"

(d) Pikachu by SUR (New Loss)

```python
def Cos_loss(llama_embeddings, llama_embeddings_1):
    clip_text_embedding1_avg = torch.mean(llama_embeddings, dim=1)
    clip_text_embedding2_avg = torch.mean(llama_embeddings_1, dim=1)


    # Normalize the embeddings
    clip_text_embedding1_normalized = torch.nn.functional.normalize(clip_text_embedding1_avg, p=2, dim=1)
    clip_text_embedding2_normalized = torch.nn.functional.normalize(clip_text_embedding2_avg, p=2, dim=1)

    # Calculate cosine similarity loss
    similarity_loss = 1-cosine_similarity(clip_text_embedding1_avg, clip_text_embedding2_avg)
    return similarity_loss
```

The above function tries to compute the "distance" between two embeddings by calculating the cosine embedding loss after normalizing over all 77 words.

10 ELLA Adapter

During our project, we encountered an intriguing corporate research endeavor by Tencent[2], introducing innovative concepts distinct from conventional methods. They propose an adapter that maps text embedded with powerful models like T5, utilizing custom adapters to generate conditioning for conditional U-Net. In contrast to the SUR adapter, their plan involves replacing the entire CLIP encoder with a custom pipeline and introducing a novel adapter named Timestep-Aware Semantic Connector. This adapter dynamically extracts timestep-dependent conditions from the LLM to condition the U-Net network, providing varied conditioning throughout the denoising process. This fine-tuned control empowers the text-to-image model to produce superior results. Majority of diffusion models currently rely on CLIP as their text encoder, limiting their ability to comprehend dense prompts. In this paper, we introduce the Efficient Large Language Model Adapter (ELLA), enhancing Semantic Understanding and Reasoning (SUR) capabilities by integrating powerful LLMs without needing to train either the U-Net or the LLM.

(a) "An aristocratic girl in medieval finery and a headdress of bright feathers drinking afternoon tea"

(b) Our ELLA Results

(c) "Three fluffy white kittens playing with a ball of yarn on a bright green carpet"

(d) Our ELLA Results

(e) "A vibrant butterfly with iridescent wings in shades of blue, green, and purple, perched on a bright pink flower"

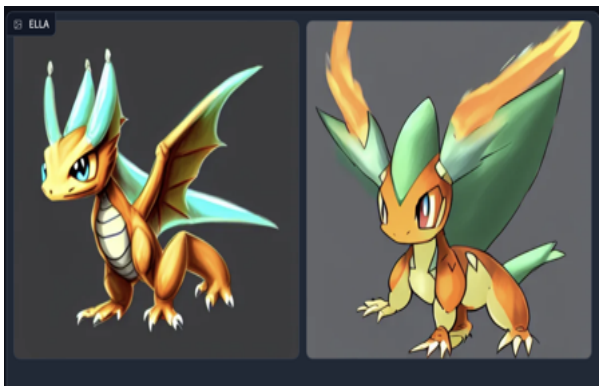
(f) Our ELLA Results

Figure 17: Analysis of ELLA on Complex Prompts

[2]Developers of PUBG

From the above figure, we observe that not only do all of the images generated by the prompts become animated (f) exactly looks like the Pokemon Butterfree!), but also the semantic understanding of the model enhances significantly. In the original diffusion model, the action of drinking tea is absent in (a), a counting error and the action of playing with yarn are missing in (c), but these errors are rectified in our results.

(a) Pikachu by ELLA



(b) Charizard by ELLA

(c) Greninja by ELLA

Figure 18: Pokemon Generation by Ella