Exploring the Latent Space of CLIP

Philipp Traber

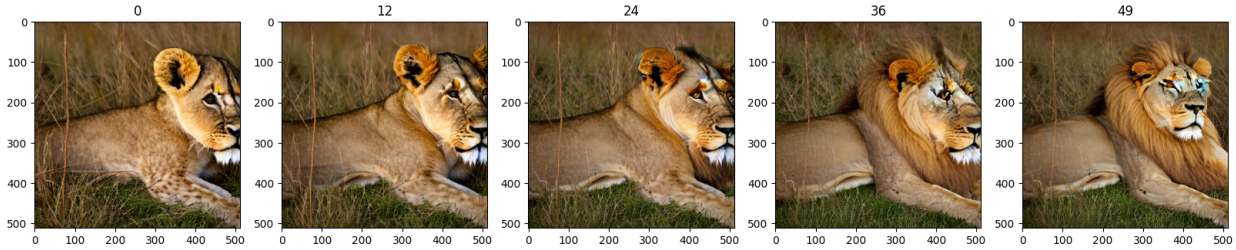

Fig. 1: Interpolated CLIP embeddings used to guide Stable Diffusion.
Initial prompts are 'An artistic photograph of a centered *age* in laying in the Savannah' where *age* is 'young lion baby cub' or 'adult'

Abstract— Connecting text and image modalities in machine learning poses a significant challenge due to the inherent differences in their properties. Contrastive Language-Image Pretraining (CLIP) addresses this challenge by training a shared latent space between text and images, thereby unlocking new possibilities. Comprehending this multimodal latent space is essential for various applications, such as text-guided image editing. This project focuses on a comprehensive exploration of the latent space of CLIP through the examination of encoded text prompts. By employing dimensionality reduction techniques such as PCA and UMAP, we gain valuable insights into the relationships among encoded prompts. Our experiments involving linear interpolation reveal that even between related prompts, meaningful content is preserved in the encoded representations. Furthermore, an analysis of the changes occurring when modifying specific properties of prompts suggests the potential to disentangle the latent space of CLIP.
The code for this project can be found at https://github.com/traberph/exploringCLIP

Index Terms—CLIP, Stable Diffusion, Latent Space, Disentanglement, Linear Interpolation

1 Introduction

In recent times, the domain of Artificial Intelligence (AI) has witnessed a notable increase in attention, leading to swift progress and noteworthy developments. Numerous beneficial tasks have been explored, such as image classification, object detection, and image segmentation within the domain of computer vision. Similarly, in the realm of natural language processing, endeavors encompass text classification, part-of-speech tagging (POS tagging), and named entity recognition (NER). A shared characteristic among these examples is their confinement to a singular modality, either text or images.

Combining text and image modalities in a multimodal AI model is challenging due to the inherent differences in their data representations and structures. Textual data is sequential and symbolic, while images consist of dense pixel grids, requiring distinct processing methods. Additionally, integrating contextual understanding across modalities poses difficulties in capturing nuanced relationships between textual and visual elements, demanding advanced model architectures and extensive training datasets for effective fusion.

In the realm of computer vision tasks, Convolutional Neural Networks (CNNs) have unequivocally emerged as the preferred paradigm. In the last decade, researchers have started to use CNNs to blend text and image modalities, predicting matching words, n-grams, or entire text captions for a given image [15, 16, 37].

The introduction of the Transformer architecture has significantly enhanced the performance and capabilities of Language Models (LM) [5, 37]. Consequently, researchers have increasingly delved into exploring the integration of the Transformer model into vision tasks, recognizing its potential to augment and improve various aspects of visual processing. The fusion of Transformers and Convolutional Neural Networks (CNNs) has yielded improved performance in contrast to methodologies solely based on CNNs [4, 18, 41]. Transformer-specific tasks, such as Masked Language Modeling (MLM) are extended into Image-Conditioned Masked Language Modeling (ICMLM) [29]

In 2021 the Vision Transformer (ViT) is introduced [7] and brings a new powerful type of image encoder inspired by the transformer architecture for text. Images are split into patches analog to the text tokens in the standard transformer. Those patches can then be processed sequentially. Adopting the Transformer architecture enables the concept of pertaining where a model is pre-trained on a large-scale generic dataset and later fine-tuned to a specific task with a smaller domain-specific task. This addresses the problem of limited training data that previous approaches suffered from.

Combining the concept of ConVIRT [41] paired with the powerful new ViT encoder Raford et. al introduced Contrastive Language-Image Pretraining (CLIP) [24]. CLIP uses contrastive learning to create a multimodal latent space between a transformer-based text encoder and a ViT image encoder. Using web scraped text-image pairs for training the model can really god abstract concepts and has incredible zero-shot performance in many tasks. CLIP can be used in a broad range of tasks like image classification or text-to-image retrieval. The domain of generative text-to-image models also benefits from the success of CLIP. Its powerful embeddings are used to guide an image generation process towards a predefined text caption.

Generative Text to image models have increased their sample quality enormously over the last few years. Various Deep Learning methods are utilized to generate images and different conditioning methods are integrated to give text guidance functionality. Starting with Recurrent Neural Networks (RNN) to Generative Adversarial Networks to newer Diffusion or Transformer based approaches. Each architecture has its own chances and limitations leading to new improved models within every architecture. An important part of a text-to-image model is the guidance mechanism which aligns the generated image with a text

description. The performance of the guidance mechanism strongly influences how well the semantics of the caption is transferred to the image. Many approaches use therefor CLIP because CLIP is good at capturing and encoding the semantics of a textual description providing a good starting point for a solid guidance mechanism.

This project is inspired by the results achieved by OpenAI with DALL-E2 [26]. DALL-E2 also referred to as unCLIP, is a generative-text-to image model, which uses multiple diffusion models (DMs) to generate images from text captions. The first DM creates CLIP image embeddings from CLIP text embeddings and the second DM inverters the CLIP image encoder and generates images from the image embeddings. Interpolating CLIP embeddings enabled DALL-E2 to generate a meaningful transition from one image to another image.

In this project, the powerful embedding space of CLIP will be explored to get more insights into how it is organized and what happens in the space between embeddings. To achieve similar results like DALL-E2 we choose Stable Diffusion because it is a publicly available model that utilizes CLIP embeddings for text conditioning.

The contribution of this project is

- a reproduction of the interpolation results from the proprietary model DALLE-2 using only open-source alternatives.
- visual insights into the latent space of CLIP utilizing the dimensionality reduction techniques Principal Component Analysis (PCA) and Uniform Manifold Approximation and Projection (UMAP).
- visualizations to inspect the changes in dimensions when the text prompts are changed.

Understanding how the CLIP space is organized enables new possibilities in image generation and editing.

2 Background

In this project, the primary focus is on the CLIP embeddings and the outcomes obtained when passing modified embeddings to the stable diffusion pipeline. This section aims to provide valuable background knowledge of how CLIP and Stable Diffusion work to understand the experiments later conducted.

2.1 CLIP

In 2021 Raford et. al introduced a new multimodal neural network called CLIP [24]. The name is an acronym for Contrastive Language-Image Pre-training. The goal of the model is to train a shared latent space for a text-encoder and an image-encoder to be able to connect those concepts. The architecture of CLIP is inspired by a previous approach to align text and images in the medical domain Contrastive Visual Representation Learning from Text (ConVIRT) [41]. OpenAI experimented with Residual Networks (ResNets) [13] and various sizes of Vision Transformers (ViTs) [7] for the image encoder. The optimal outcomes were achieved using a ViT with 307 million parameters and 24 layers. In this configuration, the input image is divided into 14 patches for sequential processing. The abbreviated name for this model is ViT-L/14@336px. The selected text encoder is a customized Transformer [36] model, altered in a manner similar to GPT-2 [25], featuring 12 layers, 8 attention heads, and a width of 512. Both text and image encoders are further equipped with an additional linear projection into the shared multimodal embedding space.

Throughout the training procedure, a technique known as contrastive learning was employed. In this method, the image encoder is supplied with a batch of images, and the text encoder is provided with the corresponding text captions. The optimization objective of the model is to maximize the cosine similarity between correct text-image pairs, while concurrently minimizing the cosine similarity for all non-correct pairs. This contrastive learning objective has proven to outperform a predictive objective [35].

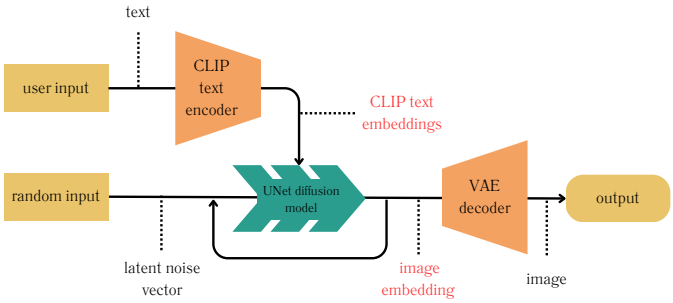

Fig. 2: Stable Diffusion Pipeline

After pre-training, both encoders have learned to map their inputs into the shared latent space. This capability enables the model to perform various tasks, including zero-shot image classification. In this scenario, the model is presented with an image and potential labels, and it selects the label whose text embedding exhibits the highest cosine similarity with the image embedding.

2.2 Stable Diffusion

In 2015 Sohl-Dickstein et al. [32] proposed a new Deep Learning approach, inspired by Non-equilibrium Thermodynamics. During the training process, the model is provided with data that is increasingly destroyed by noise until the original distribution is lost. This enables the model to learn how to reverse the destruction of the data and generate a sample from the trained distribution by only looking at random noise. The learned process is called a reverse diffusion process and can be applied to a broad range of data. Ho et al. [14] and Song et al. [33] successfully improved and adopted this approach to generate images.

Rombach et al. introduced the Latent Diffusion Model (LDM) [27], forming the foundation of Stable Diffusion. By relocating the reverse diffusion process from pixel space to a latent space of an autoencoder (AE) the computational demands are reduced significantly. The AE is trained for perceptual image compression, preserving the image semantics while simplifying the overall complexity of the space. To generate images, a DM following the U-Net [28] architecture is trained with the aim to denoise randomly sampled latents from the AE. The AE decoder can transform the denoised latents back to an image in pixel space.

Through additional modifications, the U-Net gains the capability for conditioning input, making Stable Diffusion a versatile model for text-to-image and image-to-image tasks. One application involves conditioning Stable Diffusion on CLIP text embeddings, forming the default Stable Diffusion text-to-image pipeline, visualized in Figure 2. The conditioning of the image generation process on CLIP embeddings also provides the opportunity to gain visual insights into the information encoded by the used CLIP embeddings. This makes Stable Diffusion a valuable tool for this project.

3 Related Work

This section covers topics related to the experiments conducted in this project. The modification of the CLIP embeddings aims to edit images towards a desired caption. Before the publication of CLIP, there were already existing approaches with similar objectives. subsection 3.1 gives an introduction to this domain.

This section is dedicated to discussing topics related to the experiments carried out in this project. Specifically, the modification of the multimodal CLIP embeddings is explored as a means to enhance text-guided image editing. Approaches modifying images controlled by text are covered in subsection 3.1.

Moreover, in this project, CLIP is utilized to guide Stable Diffusion. Additionally, there are other generative models that make use of CLIP

for text guidance. They are covered in subsection 3.2.

3.1 Text-Guided Image Editing

To edit images, one needs either the appropriate software and expertise or the readiness to hire a graphic designer, involving potential costs. The concept of text-guided image editing strives for models that enable image modification simply by describing the desired changes, promoting efficiency and cost-effectiveness.

Early approaches from 2017 to 2019 apply GANs to edit images with language instructions. However, these models were constrained to specific domains determined by the training dataset [6,12,19,21,45]. In 2020, the OpenEdit model, proposed by Liu et al. [17], was introduced. This model is not confined to a particular domain and exhibits the ability to make simple alterations to images, such as modifying the material or color of an object.

Muse [1] is a text-to-image transformer model that has mask-free editing capabilities. While several text-to-image models possess features like inpainting, which typically require an additional mask, Muse stands out for its ability to alter aspects of images solely based on textual descriptions, eliminating the need for an extra mask in the editing process.

Custom-Edit [2] applies techniques like Stochastic Differential Editing [20] combined with Stable Diffusion [27] to get a powerful model. Provided with a few reference images, Custom-Edit can generate new images based on a text description and the provided references.

3.2 Text-to-Image Synthesis and CLIP

The introduction of CLIP has significantly enriched the domain of text-to-image synthesis. By incorporating CLIP into other models, the challenges associated with aligning text and images are minimized, as CLIP has already aligned the latent space, and this capability can be inherited. In this chapter, generative text-to-image models are presented using CLIP.

3.2.1 Generative Adversarial Nets (GANs)

GANs [11] consist of a generator and a discriminator with an adversarial objective. The generator produces images, and the discriminator distinguishes between real and generated images, determining their authenticity. Beyond visual content, GANs have showcased text-to-image capabilities for fixed domains, transforming textual descriptions into realistic visual representations [23, 38–40, 44]. Integrating CLIP into GANs opened new possibilities for domain-independent text-to-image approaches.

CLIP-Guided Generative Latent Space Search (CLIP-GLaSS) [10] is a model to generate images from text captions. Therefore a Genetic Algorithm is paired with a GAN to search for the image that gets the highest alignment with the CLIP text embedding.

The Vector Quantised Generative Adversarial Network (VQGAN) [8] is a powerful model for generating images. With VQGAN-CLIP [3] a VQGAN is used to generate an image and a CLIP model is used to provide feedback on how to modify the initial input for the VQGAN to better match a text caption. This task is repeated until the generated image aligns with the text caption. In this approach, no extra training is needed because of the usage of two individually pre-trained models.

Both LAFITE [43] and LAFITE2 [42] integrate CLIP into a GAN approach. They use CLIP to generate pseudo text-features to be able to train LAFITE on image-only datasets.

Generative Adversarial CLIPs (GALIP) [34] seamlessly incorporate CLIP into both the generator and discriminator components of a GAN.

StyleGAN-T [30] substitutes the class conditioning features found in StyleGAN-XL [31] with a CLIP text embedding conditioning mechanism.

3.2.2 Diffusion Models(DM)

Diffusion Models described in subsection 2.2 about Stable Diffusion, are also capable of text-to-image synthesis

GLIDE [22] is a 3.5 billion parameter DM that employs both classifier-free guidance and CLIP guidance to generate images. To enhance results, a custom CLIP model was trained specifically to better handle noise, a critical aspect for DMs.

DALL-E2 [26] is a two-stage model for text-to-image synthesis. A prior is used to generate CLIP image embeddings from text captions and a decoder generates images from the CLIP image embeddings. This model is also referred to as unCLIP because it aims to invert the CLIP encoders

Stable Diffusion [27] is a more efficient approach because the DM is applied in the latent space of a pretrained autoencoder. For a comprehensive understanding of Stable Diffusion, please refer to the detailed description in Section subsection 2.2.

4 Methodology

In this section, the experiments we conducted are explained and the results are discussed. The first experiments aim to plot the embeddings. To be able to plot the embeddings the high-dimensional space has to be reduced to two dimensions. For this task, the dimensionality reduction techniques PCA and UMAP are employed. To get an insight into the space between embeddings of text captions, interpolation is used. Additionally, the changes in dimensions of an altering text prompt are examined to identify what is changing.

4.1 Dimensionality Reduction

The initial phase of our project involved gaining familiarity with the latent space of CLIP. To achieve this, we employed dimensionality reduction techniques to gain insights into the high-dimensional embeddings. Using PCA, we reduced the 768 dimensions of CLIP text embeddings to 2 dimensions, allowing the creation of a scatterplot for prompts encoded by the CLIP text encoder. As illustrated in Figure 4, the scatterplot vividly demonstrates that similar prompts are closely clustered, while the two unrelated prompts are positioned at the opposite end of the plot.

This PCA mapping is only fitted to the few embeddings shown in the plot so it does not represent the whole CLIP space. To obtain a more representative projection of the entire CLIP space, we fitted a UMAP model to ten thousand CLIP embeddings created by randomly sampling captions from the DataComp-1B dataset [9] and encoding them with the CLIP text encoder. Given the curse of dimensionality, cosine similarity was employed as the distance measure, to deal with the challenges of high-dimensional data. Cosine similarity is particularly effective in such scenarios because it compares the direction of two vectors, providing a robust measure of similarity that is less sensitive to the dimensionality of the data. This approach helps mitigate the adverse effects of sparse space, making it a suitable choice for comparing and analyzing high-dimensional embeddings. In Figure 5, the cosine similarities of the CLIP-encoded prompts from the previous example (Figure 4) are showcased. Once again, it is evident that related prompts exhibit higher similarity compared to the other two prompts.

In Figure 6, we present a visualization of prompts created by pairing a set of objects ({monkey, child, car}) with different locations ({on the moon, in the jungle, at the beach}). These prompts have been embedded and reduced using the previously fitted UMAP mapping. The resulting plot provides a representation of the relationships and proximity between these prompts in the reduced UMAP space. The analysis of the UMAP plot reveals that prompts involving the same objects are positioned closely to each other, indicating a clustering effect based on object type. Furthermore, the proximity of the "monkey" and "child" prompts is notably closer than that of the "car" prompts, suggesting a stronger semantic association between the former pair. Additionally,

Fig. 3: Prompts from left to right: 'A picture of an all-terrain vehicle', 'A picture of a race car', 'A picture of a race car in the mountains' Showing implicit encoded locations (left two pictures) and explicit encoded locations (right picture)

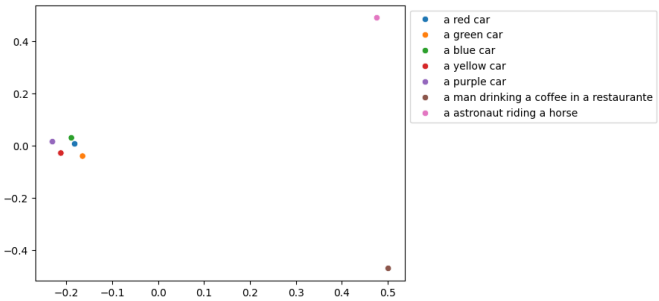

Fig. 4: PCA applied to CLIP encoded text captions

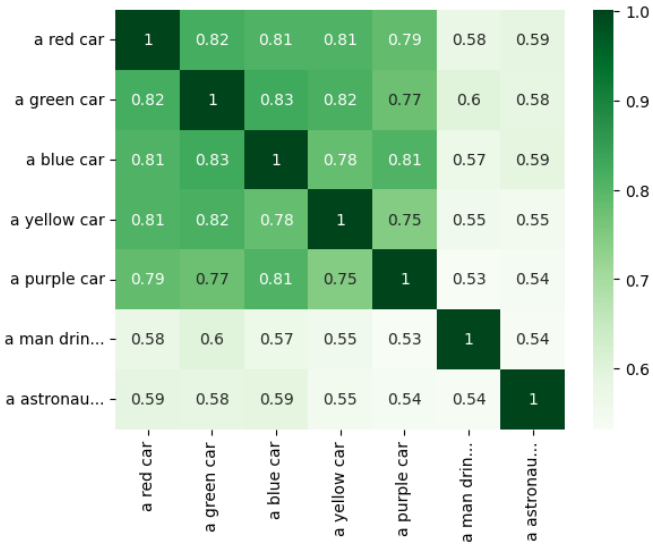

Fig. 5: cosine similarity of CLIP encoded text captions

the location "on the moon" exhibits a higher offset compared to other locations such as "in the jungle" or "at the beach," suggesting that the unusual location has a distinct impact on the overall embedding arrangement.

4.2 Interpolation

To gain insight into the relationships among text embeddings and the space between them, we employed interpolation. Initially, we generated two CLIP embeddings as start and end points by inputting two manually selected prompts into the CLIP text encoder. Using linear interpolation, we calculated 48 additional embeddings between the two initial ones. To visually examine these embeddings, we employed stable diffusion to generate images with a fixed seed and conditioned the U-Net on the interpolated CLIP embeddings. Our experiments focused on prompts where only a single variable of a fixed object changed, such as the location/background or color of an object.

It is crucial to consider implicitly encoded information, where details such as color, shape, or location may not be explicitly mentioned but are inferred from other attributes in the prompt. In Figure 3, three car images are depicted. In the case of the left two cars, the location is not explicitly specified, but CLIP deduces it based on the type of car. For instance, a race car is more likely to be associated with a race track, while an all-terrain vehicle is likely found in mountainous terrain. However, explicitly specifying the location also allows for placing a race car in a mountainous environment. The images contain various other implicitly encoded attributes, such as a shared perspective among all three pictures.

The optimal interpolation results are obtained when the prompts are formulated with precision to eliminate the introduced bias of CLIP. Fixing all the attributes gives CLIP less freedom to change those attributes

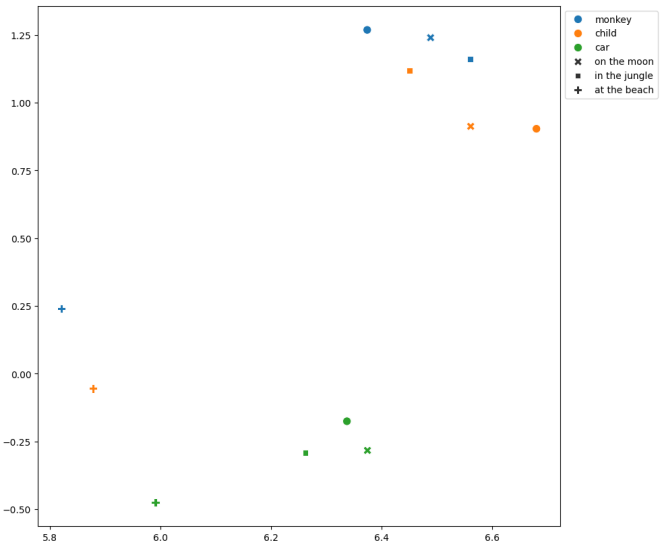

Fig. 6: UMAP applied to CLIP encoded text captions

during the interpolation leading to a smoother transition.

In Figure 7, various steps of the interpolation between the prompts *"A car in the jungle"* and *"A car at the beach"* are depicted. Due to the prompts being relatively brief and imprecise, the interpolation produces a less smooth transition. Notably, aspects like the color and shape of the car undergo substantial changes. While the overall outcome in Figure 7 may not seem promising, there are still meaningful images generated in this experiment. Figure 8 illustrates interpolation steps twenty-two and twenty-three out of fifty. These images exhibit characteristics of both initial prompts, with the background resembling a jungle from the first prompt and resembling a beach from the second prompt.

In Figure 1, the prompts are more precisely formulated to prevent unwanted changes. The objective is to observe the aging process of a lion, so the only difference in the prompts is the age of the lion. Consequently, the interpolation result exhibits a significantly smoother transition.

Future work could include additional interpolation methods like spherical interpolation.

4.3 Changes in Dimensions

The next experiments aim to identify dimensions encoding changes in specific properties of an object. If it is possible to isolate such dimensions, it enables the user to modify CLIP embeddings towards a desired direction without changing other fixed attributes.

The initial step is to generate a set of prompts to analyze. In our case, each prompt consists of an object paired with either a location or a color. These prompts are then embedded using the CLIP text encoder. to gain insight into the changes within the embeddings, they get pairwise subtracted and the absolute differences are calculated. The plot in Figure 13 illustrates the compared prompts on the diagonal. The line plots depict differences across all dimensions, and the bar chart showcases the ten dimensions with the highest differences. To enhance the readability of the visualization, the dimensions are organized based on the magnitude of change, from high to low.

A steeply descending curve in the line plot suggests that only a few dimensions encode the change in the prompt, while a flat curve indicates that a lot of dimensions are modified to encode the change. Examining the example in Figure 13 reveals that nearly all dimensions have changed, but some have experienced higher changes. The bar chart conveys that there is a significant overlap in the dimensions exhibiting the highest changes across various variations of the prompts.

To gain a deeper understanding of the overlap in changed dimensions, I squared the differences and summed them by dimension for a set of prompts. The prompts used in Figure 9 are *"A monkey "* paired with seven different locations like *"in the jungle"* or unusual locations like *"on the moon"*. The steep decline in the line plot confirms the substantial overlap in changed dimensions.

Changing the object but not the locations results in a slightly different set of dimensions. This suggests that the encoding of an object's location is dependent on the object itself. Figure 10 depicts the results of substituting "a monkey" with "a child" in the prompts. Eight out of the top twenty dimensions with the most significant changes overlap.

Conducting similar experiments with objects paired with colors yields consistent results. Figure 11 depicts the outcome of the prompts *"A {color} car"*, while Figure 12 illustrates the result of the prompts *"A {color} pullover"*. In this case, thirteen out of the twenty dimensions with the highest changes overlap.

In future work, it may be worthwhile to explore the creation of linear combinations derived from the combined results. Applying techniques such as PCA could further refine this process. Identified vectors could then be added to text embeddings, and the outcome could be examined using stable diffusion. If the results from this approach continue to show promise, the vector could be incorporated into image embeddings. By combining the text and image encoders with the vector arithmetic and Stable Diffusion a new text-guided image editing pipeline could be developed.

tool	version
python	3.9.7
cuda	12.2
pytorch	2.1.1
diffusers	0.24.0
transformers	4.29.2

Table 1: Important tools used

5 Environment

This chapter serves as a comprehensive exploration of the computing environment employed in this project, shedding light on key aspects crucial for transparency and reputability. The intricate interplay between hardware specifications, programming language choices, and the utilization of additional software and libraries is examined.

5.1 Hardware - BwUniCluster 2.0

The models used in this project have a high demand for computing power during the inference. Moreover, machine learning models like CLIP benefit from parallel computing, so conducting inference on a graphics processing unit (GPU) or a tensor processing unit (TPU) can substantially accelerate the model. Standard consumer GPUs often lack sufficient memory to handle inference with larger models and higher batch sizes. To meet these demands, I leverage a remote server within the BwUniCluster2.0, equipped with a Tesla V100-SXM2 featuring 32GB of memory.

5.2 Programming Language - Python

The programming language of choice for this project is Python, a popular language in the domains of Data Science and Machine Learning. Python's extensive libraries and tools facilitate a fast and efficient prototyping workflow.

Furthermore, the open-source version of CLIP released by OpenAI is implemented in Python, ensuring seamless integration into my project.

5.3 Computing Platform - Jupyter

Utilizing Jupyter Notebooks enables the execution of code snippets interactively, providing a fast and iterative prototyping workflow. This is particularly crucial for this project, as it allows caching intermediate results of the models. This caching mechanism proves invaluable, especially when working with large batch sizes, significantly saving time during development and debugging.

5.4 Libraries - Huggingface

Hugging Face has experienced significant growth over the last few years, emerging as the primary hub for a wide range of machine learning resources, including datasets, pretrained models, and libraries. This accumulation of resources builds a highly user-friendly ecosystem.

Within this ecosystem, the *transformers* library facilitates the loading of CLIP weights and components. The *diffusers* library allows seamless integration of the stable diffusion pipeline, while the *datasets* library provides access to valuable training data. These libraries leverage PyTorch, enabling me to inspect and modify the embeddings.

So in conclusion all decisions regarding the project setup aim to build an efficient and reproducible work environment, allowing the user to concentrate fully on conducting experiments. Versions used in this project can be found in Table 1.

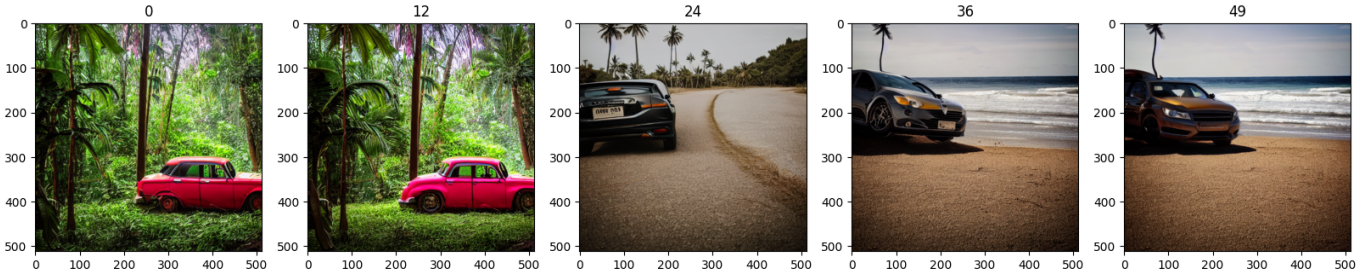

Fig. 7: Interpolating between a car in the jungle and a car at the beach - overview

Fig. 8: Interpolating between a car in the jungle and a car at the beach - steps 22 and 23

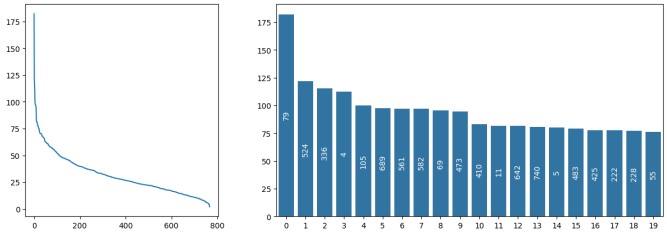

Fig. 9: Summed up differences in dimensions - monkey locations

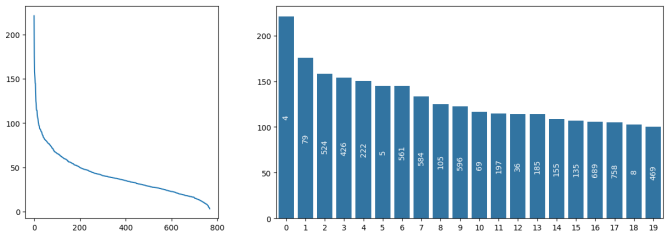

Fig. 10: Summed up differences in dimensions - child locations

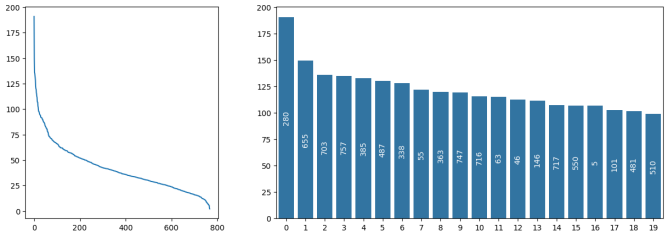

Fig. 11: Summed up differences in dimensions - car color

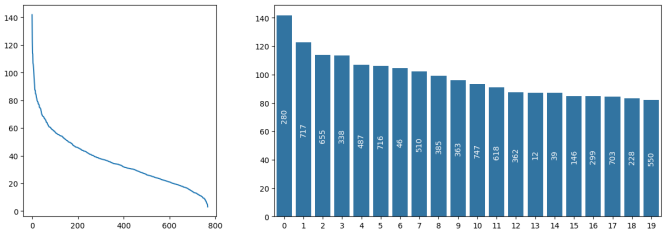

Fig. 12: Summed up differences in dimensions - pullover color

6 Conclusion

In conclusion, the investigation into the embedding space of CLIP has revealed its remarkable capability to represent a wide range of concepts and compositions.

Dimensionality reduction techniques, such as PCA and UMAP, facilitated the creation of 2-dimensional plots of the embedding space, providing visual insights. It could be observed that closely related captions are also closer in the embedding space.

During the exploration of interpolations within this space, we observed that even the areas between embeddings often encode meaningful visual content. Precision in the initial prompts constrains CLIP more, reducing unwanted changes resulting in smoother interpolations with fewer abrupt changes in the generated images.

Furthermore, the analysis of changes in dimensions indicates that it is possible to isolate a set of dimensions that encodes a specific attribute of an object in a picture. Creating a linear combination of the important dimensions could improve CLIP-guided generative AIs with fine adjustments of the generated images.

Continuing future work, exploring alternative interpolation techniques, and making further modifications to the embeddings could be pursued The adjustment of image embeddings, informed by the insights gained in Section subsection 4.3, introduces exciting possibilities for advancing text-guided image editing.

References

[1] H. Chang, H. Zhang, J. Barber, A. J. Maschinot, J. Lezama, L. Jiang, M.-H. Yang, K. Murphy, W. T. Freeman, M. Rubinstein, Y. Li, and D. Krishnan. Muse: Text-To-Image Generation via Masked Generative Transformers, Jan. 2023. arXiv:2301.00704 [cs]. doi: 10.48550/arXiv.2301.00704

[2] J. Choi, Y. Choi, Y. Kim, J. Kim, and S. Yoon. Custom-Edit: Text-Guided Image Editing with Customized Diffusion Models, May 2023. arXiv:2305.15779 [cs].

[3] K. Crowson, S. Biderman, D. Kornis, D. Stander, E. Hallahan, L. Castricato, and E. Raff. VQGAN-CLIP: Open Domain Image Generation and Editing with Natural Language Guidance. In S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, eds., *Computer Vision – ECCV 2022*, Lecture Notes in Computer Science, pp. 88–105. Springer Nature Switzerland, Cham, 2022. doi: 10.1007/978-3-031-19836-6_6

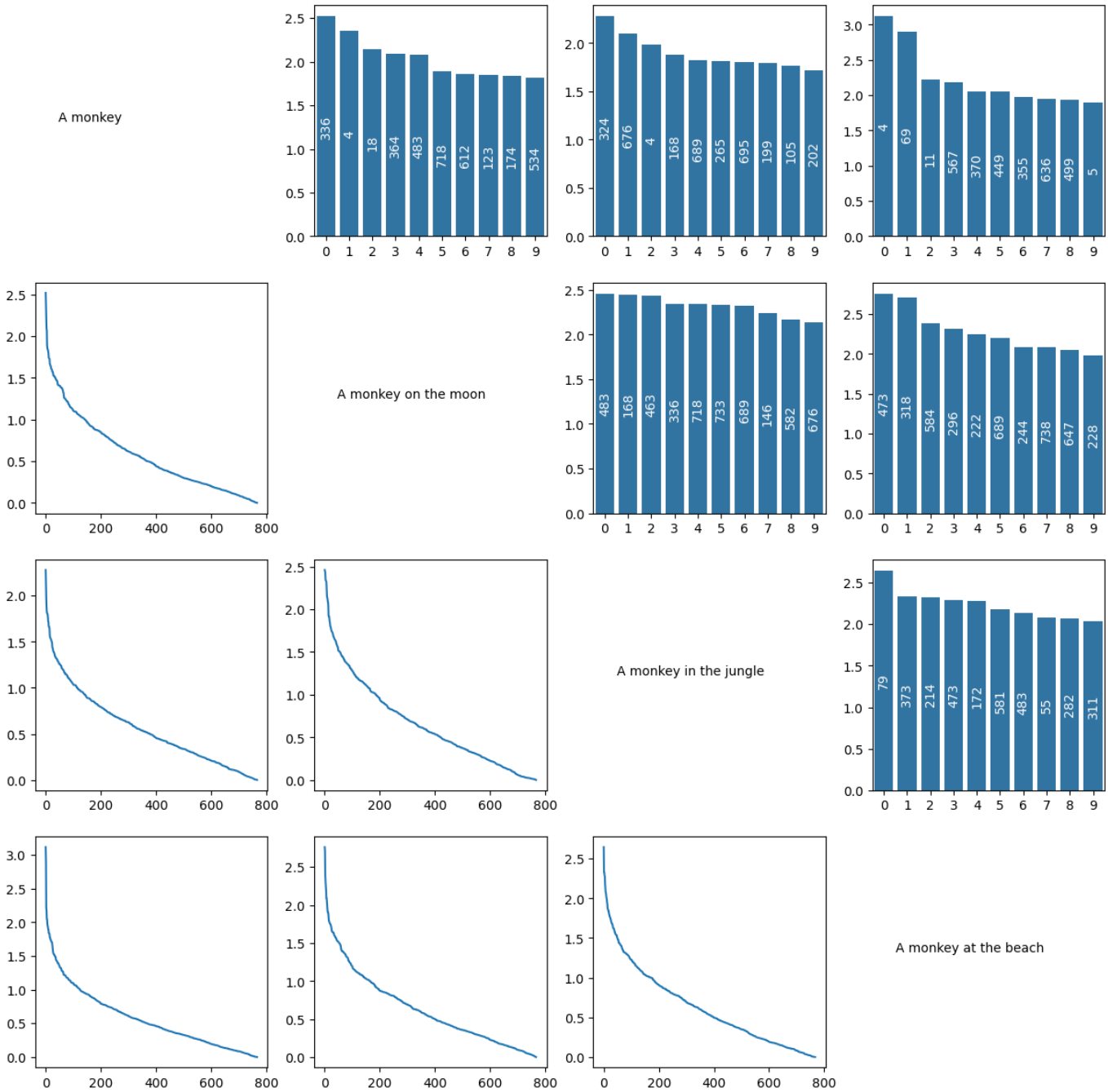

Fig. 13: Change in dimensions of CLIP embeddings

[4] K. Desai and J. Johnson. VirTex: Learning Visual Representations From Textual Annotations. pp. 11162–11173, 2021.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. arXiv:1810.04805 [cs]. doi: 10.48550/arXiv.1810.04805

[6] H. Dong, S. Yu, C. Wu, and Y. Guo. Semantic Image Synthesis via Adversarial Learning. pp. 5706–5714, 2017.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021. arXiv:2010.11929 [cs]. doi: 10.48550/arXiv.2010.11929

[8] P. Esser, R. Rombach, and B. Ommer. Taming Transformers for High-Resolution Image Synthesis. pp. 12873–12883, 2021.

[9] S. Y. Gadre, G. Ilharco, A. Fang, J. Hayase, G. Smyrnis, T. Nguyen, R. Marten, M. Wortsman, D. Ghosh, J. Zhang, E. Orgad, R. Entezari, G. Daras, S. Pratt, V. Ramanujan, Y. Bitton, K. Marathe, S. Mussmann, R. Vencu, M. Cherti, R. Krishna, P. W. Koh, O. Saukh, A. Ratner, S. Song, H. Hajishirzi, A. Farhadi, R. Beaumont, S. Oh, A. Dimakis, J. Jitsev, Y. Carmon, V. Shankar, and L. Schmidt. DataComp: In search of the next generation of multimodal datasets, Oct. 2023. arXiv:2304.14108 [cs].

[10] F. A. Galatolo, M. G. C. A. Cimino, and G. Vaglini. Generating images from caption and vice versa via CLIP-Guided Generative Latent Space Search. In *Proceedings of the International Conference on Image Processing and Vision Engineering*, pp. 166–174, 2021. arXiv:2102.01645 [cs]. doi: 10.5220/0010503701660174

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, vol. 27. Curran Associates, Inc., 2014.

[12] M. Günel, E. Erdem, and A. Erdem. Language Guided Fashion Image Manipulation with Feature-wise Transformations, Aug. 2018. arXiv:1808.04000 [cs]. doi: 10.48550/arXiv.1808.04000

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. pp. 770–778, 2016.

[14] J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851. Curran Associates, Inc., 2020.

[15] A. Joulin, L. van der Maaten, A. Jabri, and N. Vasilache. Learning Visual Features from Large Weakly Supervised Data. In B. Leibe, J. Matas, N. Sebe, and M. Welling, eds., *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pp. 67–84. Springer International Publishing, Cham, 2016. doi: 10.1007/978-3-319-46478-7_5

[16] A. Li, A. Jabri, A. Joulin, and L. van der Maaten. Learning Visual N-Grams From Web Data. pp. 4183–4192, 2017.

[17] X. Liu, Z. Lin, J. Zhang, H. Zhao, Q. Tran, X. Wang, and H. Li. Open-Edit: Open-Domain Image Manipulation with Open-Vocabulary Instructions. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds., *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pp. 89–106. Springer International Publishing, Cham, 2020. doi: 10.1007/978-3-030-58621-8_6

[18] J. Lu, D. Batra, D. Parikh, and S. Lee. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[19] X. Mao, Y. Chen, Y. Li, T. Xiong, Y. He, and H. Xue. Bilinear Representation for Language-based Image Editing Using Conditional Generative Adversarial Networks. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2047–2051. IEEE, Brighton, United Kingdom, May 2019. doi: 10.1109/ICASSP.2019.8683008

[20] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon. SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, Jan. 2022. arXiv:2108.01073 [cs]. doi: 10.48550/arXiv.2108.01073

[21] S. Nam, Y. Kim, and S. J. Kim. Text-Adaptive Generative Adversarial Networks: Manipulating Images with Natural Language. In *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018.

[22] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models, Mar. 2022. arXiv:2112.10741 [cs]. doi: 10.48550/arXiv.2112.10741

[23] T. Qiao, J. Zhang, D. Xu, and D. Tao. MirrorGAN: Learning Text-To-Image Generation by Redescription. pp. 1505–1514, 2019.

[24] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 8748–8763. PMLR, July 2021. ISSN: 2640-3498.

[25] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language Models are Unsupervised Multitask Learners. 2019.

[26] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents, Apr. 2022. arXiv:2204.06125 [cs].

[27] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-Resolution Image Synthesis With Latent Diffusion Models. pp. 10684–10695, 2022.

[28] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, eds., *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Lecture Notes in Computer Science, pp. 234–241. Springer International Publishing, Cham, 2015. doi: 10.1007/978-3-319-24574-4_28

[29] M. B. Sariyildiz, J. Perez, and D. Larlus. Learning Visual Representations with Caption Annotations. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds., *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pp. 153–170. Springer International Publishing, Cham, 2020. doi: 10.1007/978-3-030-58598-3_10

[30] A. Sauer, T. Karras, S. Laine, A. Geiger, and T. Aila. StyleGAN-T: Unlocking the Power of GANs for Fast Large-Scale Text-to-Image Synthesis, Jan. 2023. arXiv:2301.09515 [cs]. doi: 10.48550/arXiv.2301.09515

[31] A. Sauer, K. Schwarz, and A. Geiger. StyleGAN-XL: Scaling StyleGAN to Large Diverse Datasets. In *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH '22, pp. 1–10. Association for Computing Machinery, New York, NY, USA, July 2022. doi: 10.1145/3528233.3530738

[32] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 2256–2265. PMLR, June 2015. ISSN: 1938-7228.

[33] Y. Song and S. Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[34] M. Tao, B.-K. Bao, H. Tang, and C. Xu. GALIP: Generative Adversarial CLIPs for Text-to-Image Synthesis. pp. 14214–14223, 2023.

[35] Y. Tian, D. Krishnan, and P. Isola. Contrastive Multiview Coding. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds., *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pp. 776–794. Springer International Publishing, Cham, 2020. doi: 10.1007/978-3-030-58621-8_45

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[37] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and Tell: A Neural Image Caption Generator. pp. 3156–3164, 2015.

[38] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He. AttnGAN: Fine-Grained Text to Image Generation With Attentional Generative Adversarial Networks. pp. 1316–1324, 2018.

[39] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. StackGAN: Text to Photo-Realistic Image Synthesis With Stacked Generative Adversarial Networks. pp. 5907–5915, 2017.

[40] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1947–1962, Aug. 2019. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. doi: 10.1109/TPAMI.2018.2856256

[41] Y. Zhang, H. Jiang, Y. Miura, C. D. Manning, and C. P. Langlotz. Contrastive Learning of Medical Visual Representations from Paired Images and Text. In *Proceedings of the 7th Machine Learning for Healthcare Conference*, pp. 2–25. PMLR, Dec. 2022. ISSN: 2640-3498.

[42] Y. Zhou, C. Li, C. Chen, J. Gao, and J. Xu. Lafite2: Few-shot Text-to-Image Generation, Oct. 2022. arXiv:2210.14124 [cs]. doi: 10.48550/arXiv.2210.14124

[43] Y. Zhou, R. Zhang, C. Chen, C. Li, C. Tensmeyer, T. Yu, J. Gu, J. Xu, and T. Sun. Towards Language-Free Training for Text-to-Image Generation.

pp. 17907–17917, 2022.

[44] M. Zhu, P. Pan, W. Chen, and Y. Yang. DM-GAN: Dynamic Memory Generative Adversarial Networks for Text-To-Image Synthesis. pp. 5802–5810, 2019.

[45] S. Zhu, R. Urtasun, S. Fidler, D. Lin, and C. Change Loy. Be Your Own Prada: Fashion Synthesis With Structural Coherence. pp. 1680–1688, 2017.