CLIP Guided BicycleGAN

Yash Agrawal Shailesh Sridhar

Abstract—**Many image-to-image translation models perform well in implementing a one-to-one mapping between images of different domains while preserving the underlying structure. However, in many real-world scenarios, the mapping from the source to the target domain is not strictly deterministic. Traditional models like GANS, VAEs and CycleGANs have excelled in generating high quality, realistic images along with unsupervised image translations but their unimodal behavior is inadequate for tasks where multiple valid translations exist for a single input. In this project, we aim to implement BicycleGAN, which integrates the principles of CVAEGAN (reconstruction of image of domain B using paired A and latent encoding z) and CLRGAN (recovering a randomly drawn latent code z), enabling the model to produce not just one, but diverse and consistent plausible image outcomes. Next, we use Open AI's CLIP (Contrastive Language-Image Pre-Training) to guide the image generation process using text prompt. First, we rank the generator images according to the prompt and display the top n images. Secondly, we iteratively refine the image to better match the text, using backpropagation and gradient descent, starting from a random point in latent space. It not only diversifies the potential outcomes but also aligns them more closely with linguistic contexts, thereby extending the applicability and effectiveness of image-to-image translation models in complex, real-world scenarios.**

I. Complete Description of Problem

The field of image-to-image translation has witnessed significant advancements with the advent of models like Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and CycleGANs. These models excel in translating images from one domain to another while preserving the inherent structure of the input. These algorithms are widely used in applications like style transfer, photo enhancement, and domain adaptation. However, these traditional models predominantly operate under an unimodal framework. It means that they are typically designed to produce a single output for any given input image. While effective in many scenarios, this unimodal approach restricts diversity for a fixed sample or input, making them ill-suited for multimodal tasks. There is a need for frameworks that can model the complex many-to-many relationships between domains.

Simply training on diverse data is also insufficient to capture multimodal distributions spanning a wide range of semantic concepts within a domain. Additional conditioning signals and mechanisms for injecting variability are necessary. Furthermore, the generated outputs need to be guided by human-interpretable concepts and descriptions in addition to statistical similarities. This allows for fine-grained control aligned with linguistic context.

BicycleGAN provides a solution to limited diversity through its combination of variational autoencoders and generative adversarial networks. It trains models in a bidirectional way, allowing sampling of latent vectors that leads to multiple plausible outputs from the same input image. Additionally, Contrastive Language-Image Pretraining (CLIP) can match these variable images with text prompts by comparing their feature embeddings. Optimization drives the model to iteratively refine samples that score higher on relevance against the captions.

In summary, this project uniquely integrates BicycleGAN's capacity to generate varied translations and CLIP's ability to align images with text. The proposed approach can capture the uncertainty and complexity of mappings between domains. Using language guidance we can also steer the outputs to fit desired semantic descriptions. This novel approach can significantly improve image translation to handle real-world tasks requiring diversity and concept alignment.

II. Literature Review

Before the BicycleGan paper, most works in image to image translation were concerned with a single modality i.e., a one-one mapping between input image and output image. The authors of the BicycleGAN paper were more concerned with multimodal image generation, or a one to many mapping between an input image and multiple potential output image candidates.

Generative modeling involves models capable of image generation. Historically, this has been addressed through various methodologies, such as restricted Boltzmann machines and autoencoders. Variational autoencoders (VAEs) introduced a mechanism for modeling stochasticity within the network by reparametrizing latent distributions during training. Alternatively, autoregressive models were effective at capturing natural image statistics but were computationally less efficient due to their sequential prediction nature. A significant advancement in this domain was the advent of Generative Adversarial Networks (GANs), which simplified the process by mapping random inputs from an easily sampled distribution (e.g., a low-dimensional Gaussian) to generate images in a single feedforward pass. During GAN training, a discriminator network distinguished between samples produced by the generator and those from the target distribution, leading to the success of GANs in various applications. In this context, GANs demonstrated notable achievements, such as image synthesis, super-resolution, and style transfer.

Pix2Pix performs paired image translation using conditional GANs optimized with pixel-wise loss functions. This enables high-quality translation but requires supervised pairs of exact domain mappings. CycleGAN relaxes this constraint through cycle consistency losses over reconstructed inputs, allowing unpaired transfer between domains. However, deterministic approaches like Pix2Pix fail to capture uncertainty and variation in mappings. Meanwhile, CycleGAN relies on one-to-one output structuring lacking diversity.

One approach to address the issue of generating diverse outputs is to explicitly encode the various modes, or variations, in the image generation process. This is achieved by providing multiple modes as additional inputs alongside the base image. For example, past methods have utilized techniques such as color and shape information as extra conditioning. An alternative approach involves the use of a mixture of models to handle different modes. While these strategies have demonstrated success in generating multiple discrete outcomes, they face limitations in generating continuous variations.

Incorporating text prompting, CLIP relates images to text descriptions through an embedding-based similarity measure, enabling evaluation and optimization of images against language concepts. ALIGN builds on CLIP for text-guided image generation through continuous refinement. However, it is not geared for translation which is the requirement in our case.

While extensive work exists using GANs for translation and VAEs for diversity, as well as leveraging CLIP for text conditioning, at their intersection lies a gap. Integrating these approaches can model uncertainty in inter-domain mapping while aligning variability using language context. This provides new capabilities for fine-grained control over stochastic translation guided by semantic descriptions. The proposed BicycleGAN+CLIP framework explores this novel direction by synthesizing existing techniques. The approach contributes to a more robust conditional image synthesis pipeline.

III. Methodology

A. Dataset

The edge2shoes dataset is a collection distinctively divided into two domains: edge maps (Domain A) and their corresponding full-color RGB representations (Domain B). The footwear captured in the edge maps and images consists of formal shoes, casual shoes, sandals, sneakers, boots, slippers, and heels. This variety is crucial in training our model to handle multimodal translations effectively. Furthermore, each image has a resolution of 256x256 pixels along with 3 channels, although, for the purpose of optimizing training efficiency, these will be resized to a more manageable dimension. With a substantial training set encompassing approximately 49,000 images, the model is exposed to an extensive variety of real-world footwear designs and nuances. Additionally, a separate validation set of around 200 images is set aside to gauge the model's performance. A noteworthy attribute of this dataset is the absence of background distractions, ensuring that the model's focus remains undivided on the primary subject - footwear. This comprehensive dataset, with its rich design intricacies and style variations, serves as an apt foundation for our BicycleGAN's training and evaluation.

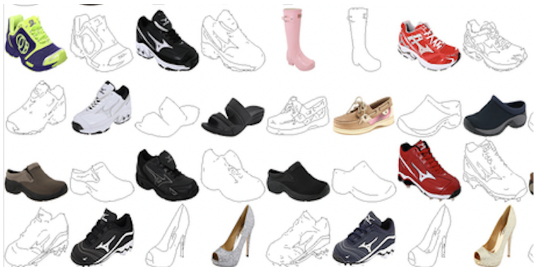
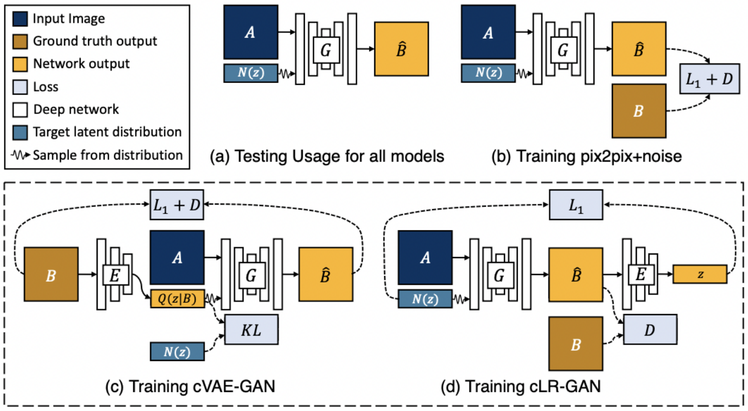


Fig. 1. Sample Sketches and Corresponding colored ground truth images from Edge2Shoes Dataset



(a) Testing Usage for all models (b) Training pix2pix+noise (c) Training cVAE-GAN (d) Training cLR-GAN

Fig. 2. High level diagram of BicycleGAN generator architecture, from the original paper

We do minimal data preprocessing in the form of normalization.

B. Architecture

- Overview As we want to introduce diversity into our possible solutions, we use one GAN, the cVAE-GAN, which learns to map from the sketch domain to the colored image domain, like one would do normally with a cycleGAN, and another GAN, the cLR-GAN. The cLR-GAN's output is passed through an encoder which maps the colored image to the latent space, z. This introduces bijectivity between the latent space and images, and results in different latent vectors being mapped to different possible outputs. This directly introduces diversity
- Generator While the schematic may result in the belief that cVAE-GAN and cLR-GAN have two distinct generators, in truth there is only one, which we use for both.

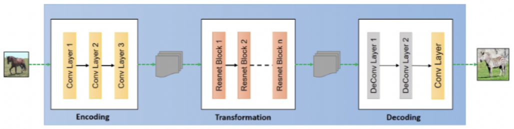


Fig. 3. High level diagram of CycleGAN generator architecture

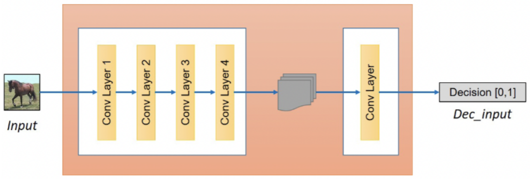


Fig. 4. High level diagram of CycleGAN discriminator architecture

Just the training scheme is different. For our architecture we use the same Resnet as the CycleGAN paper and assignment, without the instance norm.

- Discriminator We use two separate discriminators, one for each generative process i.e. (cVAE-GAN and cLR-GAN). The architecture of each discriminator is again very similar to the discriminator used in the CycleGAN paper and assignment, a 70x70 PatchGAN
- Encoder The Encoder is the same as provided in the template code, a Resnet

C. Training

1) Losses:

- The complete BicycleGAN hybrid objective and its components are given by:

$$\begin{aligned}G^*, E^* = &\arg\min G, E \arg\max D \big[L_{VAE-GAN}(G, D, E)\\&+ \lambda L_{VAE-1}(G, E) + LGAN(G, D)\\&+ \lambda_{latent} L_{latent-1}(G, E)\\&+ \lambda KLKL(E)\big]\end{aligned} \tag{1}$$

2) Hyperparameters: We used the same hyper parameters as given in the project document. We did not find noticable improvement after changing them.

D. CLIP Guided BicycleGAN

While our trained BicycleGAN gives good results on its own, we wanted to be able to guide or control the outputs with a text prompt. For example, if we wanted a white shoe for the given sketch, we wanted to be able to make the generator produce a white shoe with the prompt "White Shoe".

While we initially wanted to train with incorporation of CLIP into our training loss, we observed that making BicycleGAN converge on its own was difficult, and adding an extra CLIP based component would make it even harder to converge. Additionally, we wanted a more flexible post-hoc scheme, which would allow us to generate images matching our prompts regardless of how our model or weights.

In this light, we propose a powerful scheme for getting text-guided outputs which is both model and training agnostic and can be applied post-hoc. We compare this with a simpler but reasonably effective alternate method for doing this.

1) Naive Method: Obtaining similar images via bulk sampling: One way of using CLIP to obtain outputs matching a text prompt is by simply sampling a large enough number of n latent vectors across the sample space, getting their outputs and finding the image whose CLIP embedding is the closest to the CLIP text embedding. This Naive method works surprisingly well, as the latent space is diverse enough that if we take a large enough value of n, a few outputs will end up being close to the text prompt if the prompt is simple enough.

2) A more powerful method: Obtaining optimal latent vector via backpropagation: Our goal is simply to find the optimal latent vector for our given text prompt. As this optimal latent vector z* lies somewhere in the latent space, we hypothesized that by modeling the distance between image embedding and text embedding as a loss, we can backpropagate in the latent vector space from an arbitrary initial point with gradient descent to minimize this loss. We used a simple Adam optimizer with a learning rate of 1e-1 for this.

This worked very well, with us not only being able to obtain desired colors, but being able to find matching images for more nuanced text prompts, such as 'leather shoe' and 'metallic shoe'.

IV. Summary of Experiments

1) Generator architecture:

- Used standard UNET architecture as Generator, which involves Upsampling + Downsampling path with skip connections and Latent dimension integration. However, we observed non diverse outputs for different sampled random vectors.
- We finally ended up using a series of Residual blocks (as used in Cyclegan Generator) modified to incorporate latent integration, use Instance Normalization, and Relu activation function.

2) Instance Normalization in Generator:

- We experimented with keeping / removing InstanceNorm2d in the initial convolution block of the Generator and observed that removing it gave better results. This could be because the instance norm could be disrupting the interaction between latent dimension and input

3) Order of optimization:

- This was the most crucial step as altering orders of optimizing Encoder, Generator, Discriminator resulted in stark differences in output in terms of diversity. We tried the following orders :

- Generator → Encoder → Discriminator cVAEGAN → Discriminator cLRGAN
- Encoder → Generator → Discriminator cVAEGAN → Discriminator cLRGAN
- Encoder → Generator → Calculate cLRGAN adversarial loss → Optimize Generator → Discriminator cLRGAN → Calculate cLRGAN L1 loss → Optimize Generator

The best order was Encoder → Generator → Discriminator cVAEGAN → Discriminator cLRGAN

4) *Loss functions:*

- The way we accumulated losses also affected our performance :
  - cVAEGAN L1 + KL Divergence + cVAEGAN adversarial → cLRGAN L1 + cLRGAN adversarial → Discriminator losses
  - cVAEGAN L1 + KL Divergence + cVAEGAN adversarial → cLRGAN adversarial → Discriminator Losses → cLRGAN L1
  - cVAEGAN L1 + KL Divergence + cVAEGAN adversarial + cLRGAN adversarial + cLRGAN L1 → Discriminator Losses
  - cVAEGAN L1 + KL Divergence + cVAEGAN adversarial + cLRGAN adversarial → cLRGAN L1 → Discriminator Losses

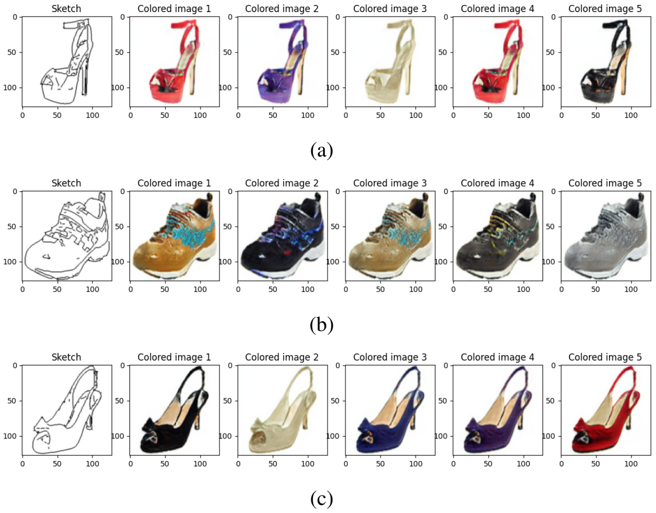
  The final accumulation in the above lists worked the best and is used in the current approach.

5) *Freezing parameters explicitly:*

- We tried freezing the generator's and encoder's parameters explicitly when we were training discriminator and vice versa. However, there was no change in performance in explicit freezing vs not freezing.

6) *Changing hyperparameters:*

- We observed mode collapse while training and hence, tried changing the lambda values for the generator and discriminator. In doing so, While we observed there were changes in output, the real reason behind non diversity was the order of optimization primarily. Hence, we ended up using the default lambda once we corrected the optimization order.
- We modified the learning rate to larger values as well. Larger learning rates of 0.002, and 0.02 gave worse results and we ended up using lr = 2e-4. We did not try much with smaller learning rates, as it would mean longer convergence time.
- We tried understanding the model behavior with small epochs of 3-5. Once we observed the model performing well, we trained on Colab Pro for 12 epochs.
- We experimented with batch sizes of 1,3 and 8. Smaller batch sizes produced non diverse outputs. This could be because a larger batch size can facilitate a more comprehensive exploration of the loss landscape. We ended up training for a batch size of 8.



(a)

(b)

(c)

Fig. 5. Some outputs of our BicycleGAN generator

7) *Incorporating MSGAN:*

- We also tried adding MSGAN loss (Mode Seeking Generative Adversarial Network) to encourage more diversity but we observed it led to non convergence. This could be due to it not being able to balance well with other losses. We ended up not using MS GAN loss

8) *Initializing weights:*

- We tried initializing weights according to different distributions for conv2D, batch norm layers but it did not make any difference and we ended up using default xavier initialization.

V. Results

A. *Qualitative Results of BicycleGAN*

Please see Figure 5 some of the qualitative results of BicycleGAN. They were diverse and photorealistic.

B. *Quantitative Results of BicycleGAN*

Photorealism:
Our FID score was:

- Real images: 51.87
- Fake images: 103.50

Diversity:
Our LPIPS score was: 0.151

Our Inception score was:

- Real Images: 5.79
- Fake images: 1.69

This implies that our model's outputs were not only photorealistic, but diverse as well.

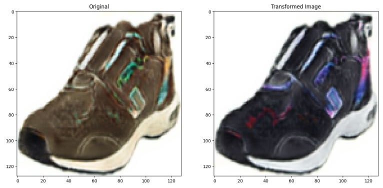


Fig. 6. original and transformed shoe with prompt ”black shoe”

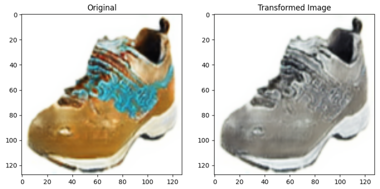


Fig. 7. Original and transformed image with prompt ”metallic shoe”

C. *Qualititative Results of CLIP-Guided BicycleGAN*

See figures 6,7 8 and 9 for examples of how powerful our simple method using CLIP can be for generating text guided outputs

VI. Future Work

While we are very happy with our results, we also see several potential directions for improvement.

First of all, while our outputs were quite diverse, we want to improve the diversity further with an MSGAN loss component or further training, perhaps with augmentation.

While our post hoc methods for matching a given text prompt worked well, we also want to experiment with a term in the loss during that uses CLIP embeddings.

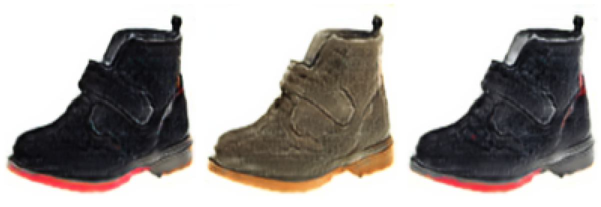

(a) (b) (c)

Fig. 8. (a) original shoe (b)with prompt ”leather shoe” (c) with prompt ”Plastic shoe”

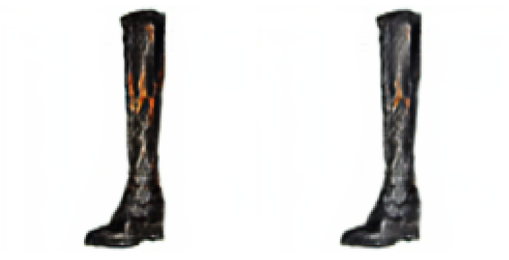

(a) (b)

Fig. 9. (a) original tall boot (b)with prompt ”silver-tinted shoe”

Finally we found that our gradient descent method for obtaining optimal latent vectors sometimes got stuck in local minima. This is probably because the latent space loss landscape has some degree of non-convexity. We can attempt to target this by training to get a smoother loss landscape (for example by using activations such as swish) or by making our optimization procedure for finding optimal latents more robust by perhaps changing the optimization method or using a larger batch size and averaging gradients

VII. References

1. Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Toward Multimodal Image-to-Image Translation.

2. Sohn, K., Lee, H., & Yan, X. (2015). Learning Structured Output Representation using Deep Conditional Generative Models.

3. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets

4. Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning Transferable Visual Models From Natural Language Supervision.

5. Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks.

6. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium.

7. Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. A. (2008). Extracting and composing robust features with denoising autoencoders.