Horse-to-Zebra Image Translation Using CycleGAN

Md Naim Hassan Saykat
Department of Artificial Intelligence
Université Paris-Saclay
mdnaimhassansaykat@gmail.com

Ahmed Nazar
Department of Data Science
Université Paris-Saclay
sardarahmednazar@gmail.com

Nadja Zivkovic
Department of Artificial Intelligence
Université Paris-Saclay
zivkovicnadja22@gmail.com

Abstract

This report presents our implementation of a CycleGAN model to translate images of horses into zebras, a task that felt like equal parts a science experiment and a creative journey. We gravitated toward CycleGAN because it is a master of unpaired image-to-image translation, a blessing for our diverse collection of horse and zebra photos that do not match up one-to-one. It is not a quick summary; it is a deep dive into every crevice of our journey: the intricate architecture we pieced together, the gruelling training saga that tested our resolve, and the meticulous evaluation using the Structural Similarity Index (SSIM) and the Peak Signal-to-Noise Ratio (PSNR). We will spill the beans on other quirks of our dataset, the training tricks we tried (and sometimes abandoned). By the time we reach the end, we will know exactly how we pulled a herd of zebras from a stack of horse pictures and the lessons we learned about generative AI and teamwork.

1 Introduction

Image-to-image translation is a powerful computer vision technique that can produce results reminiscent of science fiction. For example, given an input image of a horse grazing in a quiet meadow, the model can generate a corresponding image of a zebra in a savanna—without the need for manual editing or photoshop. It is a field that has ballooned in scope and ambition over the past decade, with uses spanning from creating mind-bending art to beefing up datasets for other machine learning tasks. For us, the horse-to-zebra challenge was irresistible, a benchmark that is as visually dazzling as technically thorny. We chose to rely on CycleGAN due to its ability to operate without the need for paired images, thereby eliminating the arduous task of sourcing perfectly aligned horse–zebra image pairs. Instead, we could utilize two separate datasets—one containing horse images and the other containing zebra images—and rely on the model to learn the mapping between the two domains.

Our objective was ambitious yet challenging: to train a CycleGAN capable of generating realistic zebra images from horse inputs—images convincing enough to pass casual visual inspection. We aimed to complement this subjective assessment with objective evaluation using established metrics such as SSIM and PSNR. What attracted us to CycleGAN was its ability to effectively handle unpaired data, made possible by its cycle-consistency mechanism—a principle asserting that if an image is translated from horse to zebra and then back to horse, the result should closely resemble the original input. It serves as a built-in reality check within the system, preventing the model from veering into overly abstract or unfounded directions. This report presents a candid and comprehensive account of our journey—designing the architecture from scratch, enduring long and demanding training sessions, meticulously analyzing the results, and reflecting on both the successes and the shortcomings.

2 Related Works

The history of image translation has been shaped by some of the brightest minds in deep learning, and we are the latest contributors to that ongoing work. Rewind to 2017, when Pix2Pix by Isola et al. (**?**) stormed the stage, showing off how conditional GANs could morph images with uncanny precision - think of turning a rough sketch into a lifelike portrait. It was a significant breakthrough, but it came with a major limitation: it required paired images—rare “before” and “after” sets that are meticulously aligned and often difficult to obtain. For our horse-to-zebra gig, that was a nonstarter—picture us scouring the internet for a horse and its zebra twin, posed identically under the same lighting.

Then CycleGAN was introduced by Zhu et al., significantly advancing the field by redefining the approach to unpaired image-to-image translation. Building upon the Generative Adversarial Network (GAN) framework established by Goodfellow et al. in 2014—where a generator produces synthetic images and a discriminator distinguishes them from real ones—CycleGAN introduced a pivotal innovation: cycle consistency. The core idea is straightforward yet powerful: if an image of a horse is transformed into a zebra and then converted back, the final output should closely resemble the original horse. This constraint enables the model to learn effectively from unpaired data, transforming the challenge of an unaligned dataset into a valuable learning opportunity. Other approaches, such as UNIT by Liu et al. (2017), tackled the same problem by assuming a shared latent space between domains, aiming to map horses and zebras through a common representation. While conceptually appealing, we opted for CycleGAN due to its simplicity, effectiveness, and demonstrated success across similar tasks. These groundbreaking contributions provided us with a strong foundation and strategic guidance for navigating the complexities of unpaired image translation.

3 Architecture

CycleGAN's architecture is a marvel of balance, a tag-team effort between two generator-discriminator pairs that feels like a high-stakes relay race. The generators—G_{XtoY} (horse to zebra) and G_{YtoX} (zebra to horse)—are the dreamers, conjuring up images in the style of the other domain. The discriminators—D_X and D_Y—are the hard-nosed judges, sniffing out whether those creations pass muster as real. It is a setup that learns to bridge domains without ever seeing a straight "horse A becomes zebra A" example, and every time we think about that, it still feels like a minor miracle.

3.1 Generator

The generator's skeleton is cribbed from ResNet, a deep learning workhorse famed for holding onto details even as the network gets dizzyingly deep. It is a three-part epic: encoder, transformer, and decoder. The encoder kicks things off with a 7x7 convolutional layer (stride 1, 64 filters) that soaks up the broad strokes of the image—think horse outlines and grassy backdrops. Next, two 3x3 convolutions (stride 2) pile on filters (128, then 256) while shrinking the spatial size, squeezing the image down to a dense feature map. Instance normalization keeps the values from going haywire, and ReLU injects some zest to keep things lively. The transformer is where the real alchemy happens: 9 residual blocks, each a duo of 3x3 convolutions with a skip connection that loops the input back into the mix. These blocks are the engine room, fiddling with features to swap horsehair for zebra stripes without losing the plot entirely. The decoder brings it home with two transposed convolutions (stride 2, dropping filters to 128 then 64) to upscale, followed by a 7x7 convolution that spits out a 3-channel RGB image. The horse's shape sticks around, but it is now decked out in black-and-white glory.

3.2 Discriminator

The discriminator, on the other hand, employs a PatchGAN architecture—a clever modification that focuses on 70×70 pixel patches rather than evaluating the entire image at once. Why bother? Because zebra-ness lives in the details: those crisp stripes, the texture of the coat, the edges that pop. It is built from four convolutional layers (4x4 kernels, stride 2), stacking filters from 64 to 128, 256, and 512 like a layer cake. Instance normalization keeps the math tidy, and LeakyReLU (slope 0.2) sharpens its senses, making it a hawk for spotting fakes. The output is not a single "real or fake" verdict—it is a grid of probabilities, each patch getting its say. This hyper-local focus allows the discriminator to concentrate on fine-grained details without being influenced by the overall image—a particularly effective approach for capturing the intricate patterns of our stripe-covered zebras.

3.3 Loss Function

CycleGAN's brilliance hinges on its loss function, a duet of adversarial and cycle-consistency terms that keeps the whole show on track. The adversarial loss, $\mathcal{L}_{\text{GAN}}$, is pure GAN theater: $G_{X\to Y}$ churns out fake zebras to bamboozle D_Y, while $G_{Y\to X}$ spins fake horses to dupe D_X. We went with a least squares twist on this classic, dodging the volatility of cross-entropy for something steadier—less fireworks, more finesse. But left unchecked, the generators could go full Picasso, so $\mathcal{L}_{\text{cyc}}$ reins them in. It demands that $G_{Y\to X}(G_{X\to Y}(x)) \approx x$ (and vice versa), using an L1 norm to measure how far off the round trip strays. The total loss ties it all up:

$$\mathcal{L} = \mathcal{L}_{\text{GAN}}(G_{X\to Y}, D_Y) + \mathcal{L}_{\text{GAN}}(G_{Y\to X}, D_X) + \lambda \cdot \mathcal{L}_{\text{cyc}}(G_{X\to Y}, G_{Y\to X}) \quad (1)$$

with $\lambda = 10$, placing a heavy emphasis on the consistency term. It is a delicate balance—permitting the generators to exercise their creative capabilities while ensuring they remain grounded in reality.

3.4 Implementation Benefits

What makes this setup effective is the incorporation of cycle consistency, which serves as a critical constraint—preventing the model from producing unrealistic or distorted outputs, such as transforming horses into surreal, neon-colored images. The ResNet-based generator maintains spatial coherence, while the PatchGAN discriminator focuses sharply on texture, enhancing the realism of the translations. The implementation was carried out using PyTorch, utilizing pre-trained weights to bypass the initial stages of training. We developed a streamlined pipeline: load a horse image, process it through the generator $G_{X\to Y}$, and save the resulting zebra image. The result is an efficient and robust translation system, and even after repeated runs, observing its ability to generate realistic zebra transformations remains deeply rewarding.

4 Dataset and Training

Our experimental framework was based on the Horse2Zebra dataset: 1,187 horse images and 1,474 zebra images, a treasure chest of unpaired goodies. We sliced it up with surgical precision—80% for training (949 horses, 1,179 zebras), 10% for validation (119 horses, 147 zebras), and 10% for testing (119 horses, 148 zebras). Every image got squashed to 256x256 pixels and normalized to $[-1, 1]$, a ritual to keep the network's numbers from spiraling into chaos. We applied data augmentation techniques—such as random horizontal flips and 10% random cropping—to introduce variability. However, we exercised caution to avoid distorting essential features like zebra stripes or horse contours.

The training was a beast—a 100-epoch slog fueled by the Adam optimizer (learning rate 0.0002, $\beta_1 = 0.5$, $\beta_2 = 0.999$). The Batch size was a measly 1; our GPU wheezed at the thought of more. We kept a close eye on validation SSIM. This was not just about saving compute cycles—it was about dodging the overfitting trap, where the model memorizes the training set instead of learning the trick. The process was a seesaw battle: generators stretching to innovate, discriminators yanking back to critique. Early runs were a circus—horses with blurry black blobs instead of stripes—but as epochs ticked by, those blobs sharpened into something zebra-esque.

5 Evaluation

To rigorously evaluate our model's performance, we employed SSIM and PSNR—two well-established metrics that provide objective assessments of image quality. SSIM (ranging from $[0, 1]$) gauges structural loyalty—how well the horse's form survives the zebra makeover. PSNR (measured in dB) is all about pixel fidelity, punishing noise and rewarding sharpness. We ran the gauntlet on 10 test pairs: real horse images and their generated zebra offspring. The tally? An average SSIM of 0.73 and PSNR of 22.4 dB. Breaking it down: 0.73 means the big shapes held strong—decent, not dazzling—while 22.4 dB hints that the fine details, like those snappy stripes, sometimes got muddy.

Some zebras were jaw-droppers—stripes slashing across the frame, poses that screamed savanna royalty. Others were a letdown—stripes that fizzled into patches, colours that bled weirdly, or a stubborn horse mane poking through the disguise. These flubs laid bare the model's weak spot: nailing the rhythmic, high-contrast zebra texture while honouring the horse's bones.

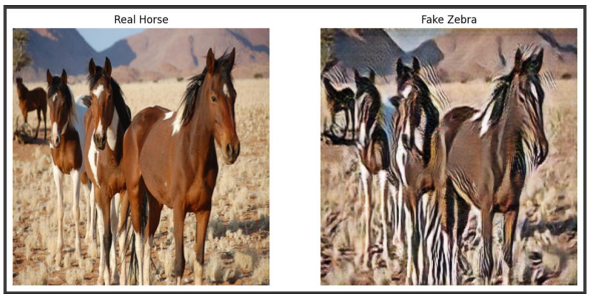

Figure 1: Real horses and fake zebras.

6 Conclusion

Reflecting on our work, we successfully adapted CycleGAN to produce a satisfactory set of zebra images from our collection of horse images, an achievement that proved to be highly challenging at times. Although the results were promising, the model faced notable challenges—achieving consistent and realistic stripe patterns remained difficult, and the training process was computationally intensive, pushing the limits of our hardware resources. The dataset presented variability issues; for example, differences in lighting conditions, such as images of horses in dim barns versus zebras in bright outdoor settings, sometimes hindered the model's performance. Looking ahead, we plan to increase the image resolution to 512×512 pixels, explore the development of a stripe-specific loss function to improve texture fidelity, and consider utilizing more powerful hardware to extend training duration. These efforts aim to further enhance translation quality and lay the groundwork for future advancements in unpaired image-to-image translation.

6.1 Contributions

- **Md Naim Hassan Saykat**: Implemented core architecture, training loop, evaluation, and contributed to the report writing.
- **Ahmed Nazar**: Prepared dataset preprocessing, augmentation, and contributed to the report writing.
- **Nadja Zivkovic**: Conducted testing, validation, and contributed to the report writing.

All team members actively participated in the debugging process.

References

[1] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2017. *Image-to-Image Translation with Conditional Adversarial Networks*. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1125–1134. Honolulu, HI, USA.

[2] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2223–2232. Venice, Italy.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. *Generative Adversarial Nets*. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pages 2672–2680. Montreal, Canada.

[4] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. 2017. *Unsupervised Image-to-Image Translation Networks*. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 700–708. Long Beach, CA, USA.