A Hybrid Adversarial Generative Approach to Generate a White Pegasus

Anonymous author

Abstract

We present an implementation and analysis of a deep generative model producing from the CIFAR-10 dataset a diverse set of winged horse images, alternatively known as Pegasi. We specifically use variational autoencoders (VAEs) to leverage the impact of a Generative Adverserial Network (GAN). We replace mean-squared error in VAE with a higher level similarity metric. This is because the element-wise measure used by VAEs do not effectively communicate differences visible to the human eye. A significant pixel-level error would not be as noticable an object transition, for example. By measuring sample similarity with a GAN's discriminator we train this in conjunction with a VAE. We find this approach produces results superior to a standard VAE when training on birds, airplanes and horses.

1 Methodology

This model combines a VAE with a GAN to learn a higher level image similarity metric more sensitive to human perception than the default element-wise metric.

1.1 Variational Autoencoders (VAEs)

The VAE (figure 1) comprises two networks: an encoder, translating data x to a latent representation z, and a decoder, attempting to reproduce the original input from the latent space representation [3].

$$z \sim Enc(x) = q(z|x) \quad \hat{x} \sim Dec(z) = q(x|z),$$

The primary concept involved in a VAE is *dimensionality reduction*. The encoder selects a small set of representative features for the input, and the decoder decompresses these, restoring the original image in a lossy manner. The VAE loss is calculated using a prior regularisation term and the reconstruction error:

$$\mathcal{L}_{\text{VAE}} = -\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{z})}{q(\boldsymbol{z} \mid \boldsymbol{x})}\right] = \mathcal{L}_{\text{llike}}^{\text{pixel}} + \mathcal{L}_{\text{prior}}$$

where

$$\begin{aligned} \mathcal{L}_{\text{llike}}^{\text{pixel}} &= -\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x} \mid \boldsymbol{z})] \\ \mathcal{L}_{\text{prior}} &= D_{\text{KL}}(q(\boldsymbol{z} \mid \boldsymbol{x}) \| p(\boldsymbol{z})) \end{aligned}$$

and D_{KL} is the Kullback-Leibler divergence, which, put simply, is a measure of difference between the probability distribution projecting the data into latent space $P(z|X)$ and its simpler estimate $Q(z|X)$. We aim to minimise this [4].

1.2 Generative adversarial networks (GANs)

A GAN (figure 2) also consists of two networks, the generator and discriminator. It aims to determine a binary classifer that best distinguishes between true training samples and fake images. As discussed, we only implement the discriminator. It determines the probability for input x being a real training image, $y = Dis(x) \in [0, 1]$.

The performance of a GAN is usually measured with minimax loss

$$\mathcal{L}_{\text{GAN}} = \log(\text{Dis}(\boldsymbol{x})) + \log(1 - \text{Dis}(\text{Gen}(\boldsymbol{z})))$$

where x is a training sample and $z \sim p(z)$ [2]. We use this as a starting point in our hybrid.

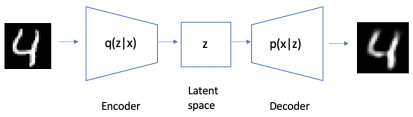

Figure 1: VAE overview

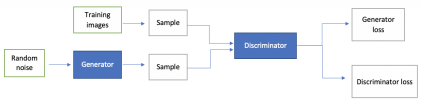

Figure 2: GAN overview

1.3 A Hybrid (VAE-GAN)

The backbone of a GAN discriminator is its ability to learn a higher level similarity metric to differentiate between real and fake images. To take advantage of this, the functionality of a GAN generator is replaced with a VAE decoder. This is possible because they both share the same parameters and map from a latent representation to a reconstructed input. Most importantly, the VAE reconstruction error term is replaced with one from the GAN discriminator (figure 3 [5]). The final loss is expressed as

$$\mathcal{L} = \mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{llike}}^{\text{Dis}_l} + \mathcal{L}_{\text{GAN}}$$

where $Dis_l(x)$ is the hidden representation of the lth layer of the discriminator [5]. We train the GAN and VAE simultaneously.

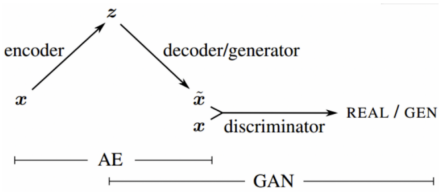

Figure 3: VAE-GAN abstraction

We filtered the dataset to horses for the pegasus body, and airplanes and birds for the wings. Airplanes were selected since we found their wings to be very prominent, and these could not be distinguished from bird wings in the final results, which arguably, is where a lack of output sharpness works in our favour. The encoder comprises of three 2D convolutional layers, a linear layer and a LeakyRelu activation function. μ and logvar are used to determine the hidden representation z. The decoder performs the reverse functionality, with the final layer using a Tanh activation function, as suggested by [6] for improved learning rates. The discriminator is composed of four 2D convolutional layers, two linear layers and a final sigmoid function so the output $\in [0, 1]$. The overall architecture is illustrated (figure 4, 5, 6).

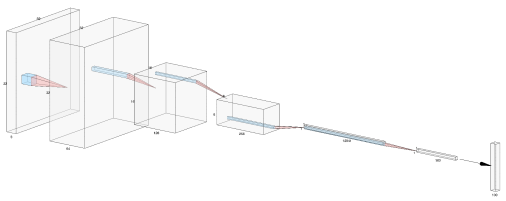

Figure 4: Encoder architecture

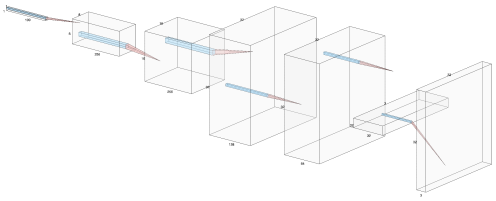

Figure 5: Decoder architecture

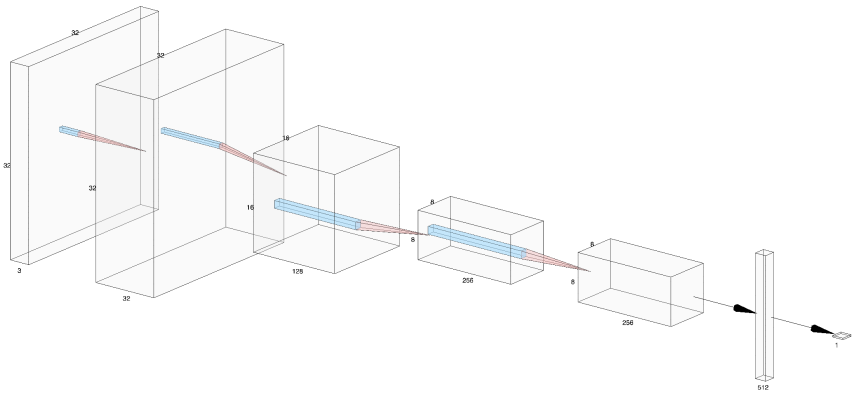

Figure 6: Discriminator architecture

We ran the system for 250 epochs, since anything less than 100 produced blurry results, as will be discussed in the following section. The VAE encoder, decoder and GAN discriminator were trained simultaneously. The discriminator would determine a latent space vector from real images, and use this and noise to create fake inputs. We aim to minimise loss, calculated in the above equation. For our final interpolation, we weighted horses and wings with a 0.65:0.35 ratio, since the horse body is the most prominent feature of a pegasus.

2 Results

Figure 7 shows a grid of results produced after training for 250 epochs. We find that more epochs improve results significantly. Anything less than 100 epochs developed unsatisfactory outputs. Refer to figure 8 for results from 20 epochs. These are too blurry to see any pegasus-resembling creature, with any stretch of the imagination. In our final batch, we can identify a diverse range of pegasi, some brown, black, and most importantly, white. In fact, we believe the best pegasus generated is in fact white (figure 9).

The diversity of horses produced and the constantly changing loss in the discriminator successfully indicates no mode collapse (figure 12).

Figure 7: 200 epochs

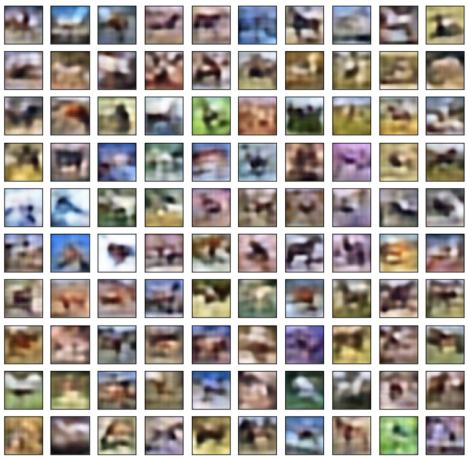

Figure 8: 20 epochs

Figure 9: A white pegasus

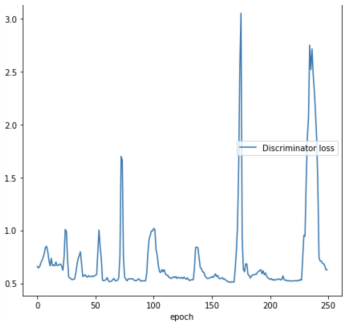

Figure 10: Discriminator loss over 250 epochs

3 Limitations

A clear limitation of the implemented method is the blurry nature of results. It is clear when comparing 20 and 250 epoch outputs that further training improves sharpness. Therefore given more time or computational power, we could train further and improve image sharpness. This leads us onto our next limitation, being the time required to train. There are three independent networks that need to be trained, an additional one more than in a VAE or GAN. Therefore, for a given time, fewer epochs will have passed and consequently, less training completed. This can be overcome with increased computational power. Additionally, outputs could have been improved with a larger dataset. The addition of planes to our wings dataset doubled the wings dataset to 12000, but this could be extended by incorporating an alternate dataset, for example STL-10. Although these have a 96 x 96 resolution, either of these datasets could be resized to the same resolution, and training performed on both. Finally, upon data analysis, we discovered the bird images either lacked wings (e.g. solely the bird's head), or had some that were not prominent. It would be interesting to see the difference in results if we initially trained on labeled birds (e.g. bird head, bird body), or manually filtered the dataset so it only contained birds with obvious or open wings. A common issue with GANs is the *vanishing gradient* problem, where the discriminator performs too well, causing the generator to fail. As evidenced in figure 10, the discriminator improves slowly overall, but loss doesn't go below 0.5, indicating a healthy balance. However, this issue should be noted. An alternate to minimax would be Wasserstein loss, which is more resistant to this issue [1].

Bonuses

This submission has a total penalty of -6 marks (a penalty), as it is trained only on CIFAR-10, and the implementation involves adversarial training.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[2] Ian J Goodfellow et al. "Generative adversarial networks". In: *arXiv preprint arXiv:1406.2661* (2014).

[3] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[4] Solomon Kullback and Richard A Leibler. "On information and sufficiency". In: *The annals of mathematical statistics* 22.1 (1951), pp. 79–86.

[5] Anders Boesen Lindbo Larsen et al. "Autoencoding beyond pixels using a learned similarity metric". In: *International conference on machine learning*. PMLR. 2016, pp. 1558–1566.

[6] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).