Movie-to-Cartoon Translation with AnimeGANplus

Naif T. AlKhunaizi
Machine Learning Department
MBZUAI
20020031@mbzuai.ac.ae

Almat S. Raskaliyev
Machine Learning Department
MBZUAI
20020029@mbzuai.ac.ae

Munachiso S. Nwadike
Computer Vision Dept.
MBZUAI
20020056@mbzuai.ac.ae

Abstract

We present our work on transforming movies into cartoons using generative adversarial networks. Our technique is to consider the movie cartoonisation task as a frame by frame instantiation of the real world photograph cartoonisation task, and to build upon existing techniques for photo cartoonisation to cartoonise each frame, before recombining them into an animation. We improve on the AnimeGAN baseline model for photo cartoonisation, by accelerating convergence using inspiration from AnimeGANv2. The resulting network is called AnimeGANplus, and produces better results than the baseline model. We train the model with the three different cartoon datasets introduced by AnimeGAN, each one with a unique style. The datasets are unpaired, which makes the applications of the model generalisable to similar use cases. We provide visual frame results for evaluation of this technique, as well as some numerical investigations with the Frechet Inception Distance (FID) metric. Finally, we deploy AnimeGANplus in converting multiple video clips to cartoons, to evaluate the practical applicability of our technique.

1 Introduction

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 1: Cartoonisation by means of AnimeGANplus: the top row consists of images of well known actors and superheroes they played in films. The middle and bottom rows respectively represent cartoonisation using the Paprika and Hayao datasets in training. Images in the middle row, such as in column (g), may appear to represent more content loss than those in the bottom. This is not an *unacceptable bug*, but a *desirable feature*. When you combine individual frames into a movie, subtle content loss improves cartoonisation quality, as the human brain fills in the gaps. This is demonstrated by one of our experiments where the Paprika style, which had more content loss, obtained more compelling cartoonisation relative to the Shinkai style[1].

Creating animations requires drawing individual frames with a level of detail that not only maintains artistic style, but also preserves narrative semantic content. This is an intensive task for humans, but our research indicates that it can also be learned by neural networks performing domain translation. Our work uses the generative adversarial network (GAN) [1], named AnimeGANplus, to perform cartoonisation of movies by treating each frame as a picture and cartoonising it (see Figure 1). From the perspective of self-supervised learning, GANs can be viewed as belonging to a set of self-supervised learning techniques known as generation based methods [2]. From the perspective of computer vision techniques, the aim of cartoon stylization is to map frame scenes from the movie domain into the cartoon domain while preserving the visual content information of the image.

We build upon AnimeGAN, a GAN constructed to convert real world images to Japanese anime, in our work. AnimeGAN [3] has the ability to learn domain mappings from unpaired training data. It does this by treating the generator as an encoder-decoder module, and using real world images to guide the generator, as a vanilla GAN would use an input noise variable. The ability of AnimeGAN to learn domain mappings from unpaired training data provides the first reason for which we build upon it. The second reason for choosing AnimeGAN is its generalised ability to combine neural style transfer with generative adversarial networks. Compared to the dedicated, GAN based approaches for real-world image cartoonisation, such as CartoonGAN [4] and ComixGAN [5], AnimeGAN is an improved architecture created for image cartoonisation. This made it our choice for research, improvement and modification.

In our project of movie-to-cartoon translation, we are faced with several challenges in adding novel improvements to AnimeGAN. Specifically, we worked to enhance the convergence time while reducing the presence of artefacts in output images. A baseline version of AnimeGAN takes more time to run and fails occasionally to maintain the content of the images. Furthermore, while other works focus on cartoonisation of images, we seek to cartoonise entire movies or videos. Our model uses extracted frames from three different anime movies made by the same artist, each with different style. Our aspirations to overcome these challenges led to the following outcomes of our research:

- We improve convergence of AnimeGANv1 by replacing the instance normalisation in its layers with layer normalisation. This is identical to what was done in AnimeGANv2, but in our case, we maintain all other aspects of the original AnimeGANv1 architecture. The resulting neural network is called AnimeGANplus.
- We cartoonise three different movies. The first is a scene from the Star Wars Episode II, Attack of the Clones, where two Jedi present themselves to the Jedi High Council. The second is a video about trip to Dubai by MBZUAI students. The third video is an official MBZUAI promotional video made for the PR department. We provide links to these video together with our report.

2 Related Work

Research works such as [6, 7] have provided GAN based solutions for pixel-to-pixel image synthesis tasks. However, these models need to be trained on paired image datasets to produce meaningful results. This may be unsuitable for image domain translation due to the difficulty of acquiring paired image datasets. To tackle this essential shortcoming, methods such as CycleGAN [8] were introduced, as capable of carrying out image-to-image translation with unpaired training datasets. CycleGAN has been recently modified into GANILLA [9], which can translate images into children's book illustrations.

Animation style transfer is a relatively new research direction in GAN based image-to-image translation. In CartoonGAN [4], a style transfer method was introduced that showed promising results for the photo cartoonisation problem. ComixGAN focused on obtaining a similar cartoonisation result, but tailored to producing images reminiscent of comic book style [5]. ComixGAN introduced a novel key frame extraction technique that picks a frame subset from the input video and outputs representation rich video context. The structure of ComixGAN's network is the same as in [4], however the authors utilised other training approaches in their model. The more lightweight AnimeGAN [3] was presented to learn the mappings from photo to cartoon domains utilising unpaired training data. In this project, we further modified AnimeGAN for use on movie translation (see Figure 2).

[1] `dubaitrip_paprika.mp4` and `dubaitrip_shinkai.mp4`

Figure 2: Eight frames from the infamous Star Wars scene of the death of Darth Vader, cartoonised using the Hayao style. The frames are extracted using the Katna keyframe extraction technique [10], and represent the highest inter-frame variance within a two minutes and eighteen seconds duration. The eight scenes are organised into two rows (top and bottom). The top half of each row shows the original extracted frames, while the bottom half of each row shows the results from AnimeGANplus.

3 Methodology

3.1 AnimeGANplus Architecture

AnimeGANplus is a GAN that consists of two CNN networks, a generator and a discriminator (see Figure 3), trained using a minimax objective function. The inputs to the generator are real world images, to approximate the movie domain. Since the cartoon images generated are constrained to bear similarities with the real world images, the generator can be considered as effectively an autoencoder. It uses inverted residual blocks (IRBs) (See Figure 4(left)), introduced by MobileNetv2, which are less computationally expensive than the residual blocks of ResNet [11]. The discriminator is a PatchGAN [6], as generated images are considered as 64×64 image patches, each of which is judged as real, or fake, individually. The discriminator uses spectral normalisation [12], which entails dividing the weight matrix in each layer by its spectral norm, to control the maximum absolute value of the derivative at each layer. This helps prevent exploding gradients and mode collapse.

AnimeGANplus uses the layer normalisation method, while AnimeGAN [3] employs the instance normalisation method. As explained in [13], both of these feature map normalisation methods are based on the following formula:

$$x_j^{out} = \frac{1}{\sigma_j}(x_j - \mu_j) \tag{1}$$

where x is a feature computed by the normalisation layer, and j is an index. This index has 4 elements when the normalisation layer's input is a 2D image, that is $j = (j_N, j_C, j_H, j_W)$, where N represents a batch index element, C represents a channel index element, and H and W represent spatial height and width index elements.

The values of σ and μ are a standard deviation and a mean, found by the formulas:

$$\mu_j = \frac{1}{m} \sum_{k \in S_j} x_k \tag{2}$$

$$\sigma_j = \sqrt{\frac{1}{m} \sum_{k \in S_j} (x_k - \mu_j)^2 + \epsilon} \tag{3}$$

where ϵ is a very small constant number, S_j is a set of feature elements for which the means and the standard deviation are computed, and m is the size of this set. The aforementioned feature map normalisation methods differ in how the set S_j is designed. The set is defined for the layer

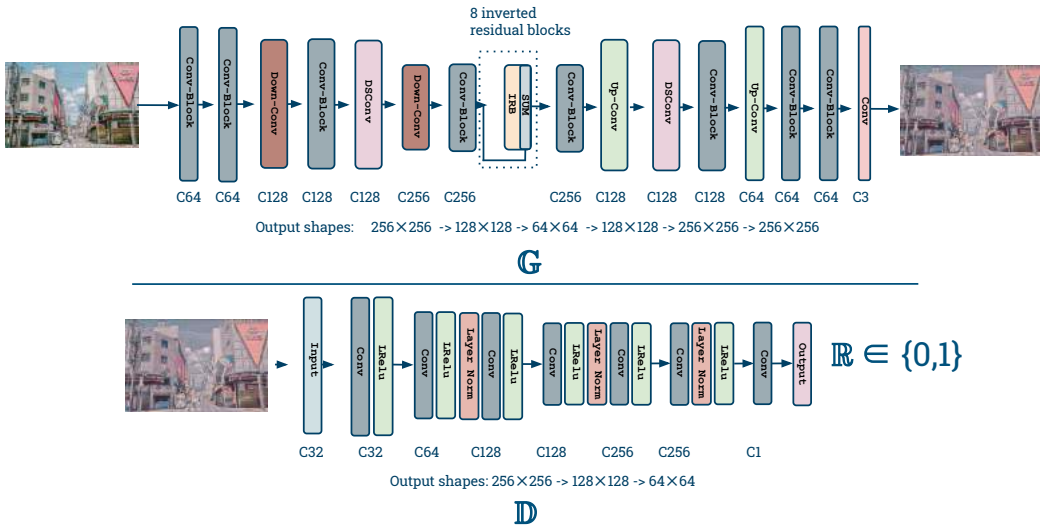


Figure 3: Diagrams of the AnimeGANplus generator G and discriminator D. The generator further consists of the special Conv-Block, DSConv, and inverted residual blocks, all arranged in an encoder-decoder structure. Notably, the last convolution in the generator does not involve an activation function. The discriminator is a 64×64 PatchGAN, using layer normalisation instead of instance normalisation. 'C' denotes the number of output channels in each block or layer. We also provide the output shapes of each layer's feature maps.

normalisation method as $S_j = \{k|k_N = j_N\}$, which means that layer norm computes the mean and the standard deviation along the index elements (C, H, W) for every sample in the batch. The set is defined for the instance normalisation method a distinct formula $S_j = \{k|k_N = j_N, k_C = j_C\}$, which means that instance norm computes the mean and the standard deviation along the index elements (H, W) for every sample and each channel.

The Figure 4 provides visual comparison of different kinds of normalisation techniques. Batch normalisation involves normalising across all activations in a batch, for a fixed channel at a time. Instance norm performs this normalisation per batch, per channel. Layer norm, however, normalises across all channels but for a fixed entry in the batch.

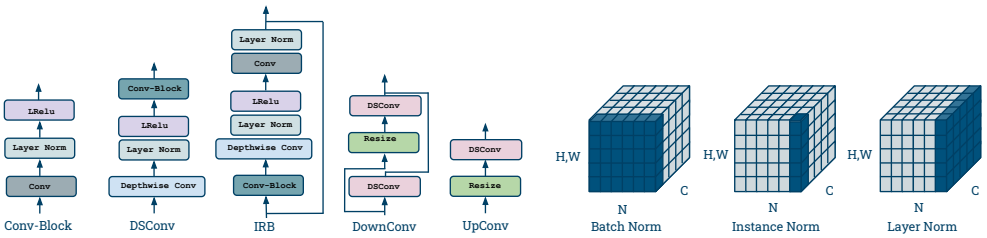
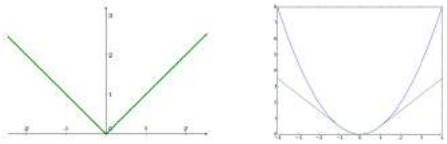


Figure 4: (Left) AnimeGANplus uses the similar building blocks to AnimeGAN: Conv-Block, DSConv, IRB, DownConv and UpConv. We utilise layer norm in three of these blocks. (Right) We explain three kinds of normalisation. H and W represent the feature map height and width. C represents the channel dimension, and N represents the batch. The total number of elements in the set for layer norm is usually larger than the corresponding number for the instance norm. That means that the layer norm provides more uniform and centered distribution of all the input features in its output, which leads to faster convergence and lower number of artefacts in the generated images of AnimeGANplus compared to AnimeGAN.

3.2 AnimeGANplus Loss Function

AnimeGANplus is trained using the four loss functions of AnimeGAN, an adversarial loss $L_{adv}(G, D)$, a content loss $L_{con}(G, D)$, a grayscale style loss $L_{gra}(G, D)$ and a color reconstruction error loss $L_{col}(G, D)$; the loss function L(G,D) can be expressed as follows:

$$L(G, D) = w_{adv}L_{adv}(G, D) + w_{con}L_{con}(G, D) + w_{gra}L_{gra}(G, D) + w_{col}L_{col}(G, D) \quad (4)$$



(a) (b)

Figure 5: (a) visualises a two dimensional L1 loss, which tends to provide uniform gradients regardless of the size of the error, while (b) shows the counterpart Huber loss, which has smaller gradients as we approach smaller values within sufficiently small neighborhood. This is important for cartoonised images, whereby we need colors to be similar, but not exactly the same, as counterpart real images.

The adversarial loss $L_{adv}(G, D)$ is responsible for the adversarial training component. The content loss $L_{con}(G, D)$ enforces the retention of content in the image after the reconstruction in the generator, and the grayscale loss causes the generated images have the saturated colors associated with animated images. The color reconstruction error loss $L_{col}(G, D)$ is responsible for keeping those colors similar to those in the original images. We weight these loss terms similarly to the original AnimeGAN [3], as it achieves a good balance of style and content. The weights were set at $w_{adv} = 300$, $w_{con} = 1.5$, $w_{gra} = 3$, w_{col} = 10. Interestingly, the content loss utilises a pretrained VGG19 model [14] for extracting high level features from generated images, whose quality are compared against those of real images using a simple L1 loss. In expanding upon these loss terms, we use the a_i to represent real animated training images, while x_i represent versions of these images which have been converted to grayscale. y_i are the grayscale versions of the images after they have been smoothed using a blurring filter. The content and grayscale loss functions can be expressed respectively as follows:

$$L_{con}(G, D) = E_{p_i \sim S_{data}(p)}[||VGG_l(p_i) - VGG_l(G(p_i))||_1] \quad (5)$$

$$L_{gra}(G, D) = E_{p_i \sim S_{data}(p), E_{x_i \sim S_{data}(x)}}[||Gram(VGG_l(G(p_i))) - Gram(VGG_l(x_i))||_1] \quad (6)$$

$$L_{col}(G, D) = E_{p_i \sim S_{data}(p)}[||Y(G(p_i))-Y(p_i)||_1+||U(G(p_i))-U(p_i)||_H+||V(G(p_i))-V(p_i)||_H] \quad (7)$$

In the equation above, '*' indicates the input to the l[th] layer of VGG19, in this case, the "conv4-4" layer of VGG19. $Gram$ is the Gram matrix of the features. [15] showed that matching of Gram matrices of real and generated image features from VGG19, would be equivalent to minimising the maximum mean discrepancy (MMD) between the feature distributions, which would contain information about their dissimilarity.

The color reconstruction loss used in AnimeGANplus, converts the original images from RGB to YUV color space. It then measures the sum of L1 loss between input and generated images in the Y channel and the Huber loss between input and generated images in the U and V channels. The intuition is that the Y, or brightness channels in the generated images should move towards a loss of zero since L1 loss tends to have more uniform gradients (see Figure 5(a)), while the U and V chrominance or color channels will be more gently modified, since Huber Loss profers lesser gradients for smaller values, within a sufficiently small neighborhood where it is quadratic (see Figure 5(b)).

Finally, the loss functions of the discriminator and the generator can be expressed as follows:

$$L(G) = w_{adv}E_{p_i \sim S_{data(p)}}[(G(p_i) - 1)^2] + w_{con}L_{con}(G, D) + w_{gra}L_{gra}(G, D) + w_{col}L_{col}(G, D) \quad (8)$$

$$L(D) = w_{adv}[E_{a_i \sim S_{data}(a)}[(D(a_i) - 1)^2] + E_{p_i \sim S_{data(p)}}[(D(G(p_i)))^2] + E_{x_i \sim S_{data(x)}}[(D(x_i))^2] + 0.1E_{y_i \sim S_{data(y)}}[(D(y_i))^2]] \quad (9)$$

Referring to prior definitions of the a_i, x_i, and y_i, it is not hard to see, with some work, that last two terms in the discriminator loss control the grayscale tinge, and prominence of edges respectively.

3.3 AnimeGANplus Dataset

(a) (b)

Figure 6: In (a) we have a frame from the Star Wars Episode III - Revenge of the Sith, where Anakin Skywalker confronts Mace Windu and the Jedi High Council. In (b) we have a counterpart frame from Star Wars Clone Wars, in which we can see Mace Windu, along with Jedi Master Yoda. Subtle differences in shape and texture distinguish these two film domains.

In an initial effort to convert Star Wars movies to animations, we gathered a dataset of Star Wars movie scene frames, extracted from YouTube (see Figure 6(a)). We extracted similar scene frames from 3D rendered Star Wars animations belonging to the "Clone Wars" series (see Figure 6(b)). However, upon initial experiments training a CycleGAN model [8] on this data, we found that we would require scenes with higher inter-frame variance. Specifically, when the images used to train a model are too similar, it contributes to mode collapse. The dataset used to train the original AnimeGAN [3], was more suitable to adopt for AnimeGANplus, due to the high variance between images, as well as the wide range colors in the real world photograph set. The dataset uses 6,656 photos from the real world to approximate the movie domain. At the same time, it uses 1792, 1650, and 1553 frames, respectively, from the anime films "The Wind Rises", "Your Name", and "Paprika" respectively, for the cartoon image domains. These films, while from the same artist, correspond to the unqiue artistic styles, and are codenamed Hayao, Shinkai, and Paprika. The resolution of the images taken from these frames is 256×256.

4 Implementation Details

Our implementation was in PyTorch. In training, we set the learning rate for the generator and discriminator to 0.00008 and 0.00016 respectively. The original AnimeGAN was trained on 100 epochs for each of the three styles. However, with AnimeGANplus, the use of layer normalisation resulted in faster convergence such that training required fewer epochs. AnimeGANplus was trained for 55 epochs for the Hayao dataset, 63 epochs for the Shinkai dataset, and 40 epochs for the Paprika dataset. We chose these epochs as representing the stages at which the model provides consistent anime texture while preserving the content of outputs. With the exception of its use of layer normalisation, AnimeGANplus has inherited most implementation aspects from AnimeGAN. For example, the model adjusts averaged brightness of the generated image to the averaged brightness of the input test image (see Figure 7). As aforement, AnimeGANplus shows impressive improvement in term of the training time compared to baseline AnimeGAN. Figure 8 shows how AnimeGANplus gets rid of the artefacts of the generated image in the earlier epochs. To obtain the individual frames from the videos, we made use of the `OpenCV` Python library. After cartoonising the individual frames, we stitch them together using the `ffmpeg` tool in Linux and `OpenCV`.

(a) (b) (c)

Figure 7: AnimeGANplus performs processing on a generated image to change its average brightness to that of the movie domain image from which it was generated. A sample image in (a) is enhanced to have the brightness of the image in (b). The result is the image in (c).

(a) (b) (c) (d)

Figure 8: The first row (top) shows results from AnimeGANv1, which uses instance normalisation, while the second row (bottom) represents results from the proposed AnimeGANplus, which replaces instance normalisation with layer normalisation. In (a), (b), (c) and (d), we have results from training epochs 5, 25, 75, and 100. Cartoonisation results with instance normalisation, continue to bear artefacts, seen in the cloud regions, until epoch 100. By contrast, as early as epoch 5, the layer normalisation achieves strong cartoonisation by epoch 25, with no artefacts. This indicates that layer normalisations allows improved convergence.

5 Experimental Results

5.1 AnimeGANplus Experiments

5.1.1 Star Wars

Our first experiment was to cartoonise frames from a scene in Star Wars Episode II, Attack of the Clones, where two Jedi present themselves to the Jedi High Council. This was done using the original AnimeGAN, trained on the Hayao dataset. The results, shown in Figure 9, were promising. The network preserved sufficient feature content to retain the identity of characters, while providing textures reminiscent of animations. The results demonstrated the initial feasibility of learning domain mappings between movies and cartoons.

When we completed AnimeGANplus, we evaluated it, much like the original AnimeGAN, using Star Wars movies. In Figure 10, we selected five iconic characters from the Star Wars cinematic universe, and cartoonised frames in which they were isolated. We performed a cropping operation to ensure that all the frames have uniform height and width. The resulting images exhibit the promotion of edges and flattening of surface features, associated with animation style. Similarly, in Figure 11, AnimeGANplus was used to generate compelling animated images for scenes with two divergent background contexts from the Star Wars universe. Outputs for the fire and ice planets of Mustafar and Hoth were compared. We observe that the darker scene frames of Mustafar made for less prominent cartoonisation effects.

Figure 9: The animated images generated by the baseline AnimeGAN, when provided Star Wars movie frames for inference. We see that the frames take on an animated, painting-like quality, which translates to distinctive texture, when stitched together into an animation sequence.

Figure 10: We isolate iconic characters from the Star Wars franchise, in order to examine the cartoonisation effects on them. Facial features that distinguish each character are maintained and edges are strengthened, while surfaces and contours are flattened to resemble cartoon drawings. Darth Vader, appearing in the middle column, comes out quite well.

Figure 11: Fire vs. Ice. Star Wars consists of vastly diverse planetary landscapes, such as Hoth, a cold planet in the top half of this figure, and Mustafar, a fiery planet in the bottom half of this figure. We contrast cartoonisation effects on these two planets by putting the original movie frames in top part of each half, and the cartoonised frames in bottom part of each half. Cartoonisation is computed using AnimeGANplus trained on the Hayao style.

5.1.2 Styles Comparison

Figure 12: We see the changes in stylistic characteristics of the cartoon effects as we modify the animation training samples in the cartoon domain. In (a) we have the original photographs, whereas in (b), (c), and (d), we have cartoonisations using Shinkai, Paprika, and Hayao styles respectively.

The objective of our first set of experiments was to convert Star Wars movie frames to corresponding animation frames. The AnimeGANplus model [3] was trained on three aforementioned cartoon datasets, with styles compared in Figure 12. All styles successfully transfer real world images into anime images. However, certain styles give better results for the image frames, than others. For example, when we cartoonised an MBZUAI promotional video, discussed in the next section, training on the Paprika style resulted in better animation effects than with Shinkai or Hayao. Furthermore, AnimeGANplus gives clearer results when original input image frames are of high quality. For low resolution images, the model will produce poor results. Hence, it is important to provide the model with high quality images during inference.

5.1.3 Comparisons with Baselines

We make a qualitative comparison of the proposed AnimeGANplus with the baseline models of AnimeGANv1 and ComixGAN (see Figure 13). We can notice that ComixGAN tends to produce darkened faces, and AnimeGANv1 generates uniformly colored images, but with lesser cartoonisation. AnimeGANplus appears more compelling in comparison to the baselines. We have also made a quantitative comparison of the original version of AnimeGAN with the three style variants of AnimeGANplus by means of Frechet Inception Distance (FID) score (see Table 1). The FID score [16] is an evaluation metric developed specifically for a comparative evaluation of GANs. It compares the statistical distribution of high level features, as per the InceptionV3 neural net, from the collection of generated images, with the corresponding distribution from real cartoon images which have been used to train the GAN. A lower FID score means that the two collections of images are more similar. Analysing the table, we can conclude that AnimeGANv1 produces more cartoonised images compared to the variant of AnimeGANplus of Hayao style, but the original model is surpassed in cartoonisation by AnimeGANplus variants of Paprika and Shinkai styles.

(a) (b) (c) (d)

Figure 13: We compare our AnimeGANplus in (d) with prior cartoonisation techniques. The original RGB images are presented in (a), while (b) represents the results of ComixGAN [5], and (c) represents the original AnimeGAN [3]. AnimeGANplus uses the same architecture as AnimeGANv1, but instead uses layer normalisation [17]. The network attains strong cartoonisation characteristics, despite its improved convergence speed.

Model	AnimeGAN	AnimeGANplus		
		Hayao	Paprika	Shinkai
Number of original images	1792	1792	1553	1650
Number of generated images	67	67	67	67
FID score	2.932	3.167	1.285	2.084

Table 1: In this table, we compare, quantitatively, the AnimeGANv1 of Hayao style, with the proposed AnimeGANplus of Hayao, Paprika and Shinkai styles. Features are extracted from corresponding real and generated images, and compared by means of the Frechet Inception Distance (FID) score. FID uses the Frechet distance, also known as the Wasserstein-2 distance, to compare extracted features from an InceptionV3 network.

5.2 Generated Cartoon Videos

The first cartoonised video which we made was a movie-to-cartoon translation on a scene from Star Wars Episode II, Attack of the Clones. We converted the individual frames using AnimeGANplus, and stitched them back together using the `ffmpeg` utility. We also added audio from the original Star Wars soundtrack to the final cut. The full video can be found online[1]. The video possesed a subtle "video game animation" effect, as was observed by members of the public to whom we showed it. Similarly, we made two cartoonised versions of a promotional video made by the MBZUAI graduate student council. These were the same videos mentioned in our introductory Figure 1. The scenes in the original version of this video were highly realistic, with vivid colors and pronounced camera motion. This made it challenging to produce cartoonisation results. However, the results with Paprika were closer to cartoons than results with Hayao, and overall, the model showed promising results. We provide a full length cartoonisation of an official MBZUAI promotional video, show in Figure 14. An interesting observation from this video was that the more closeup a video character was to the camera, the stronger the cartoonisation effects noticed on them. In other frames, the original video contained little motion, with few characters in each frame, making it easy for the model to provide convincing cartoonisation.

(a) (b) (c) (d) (e)

Figure 14: Selected frames from the MBZUAI promotional video[1] we translated into a cartoon. We observed improved cartoonisation on closeup shot segments. The limited motion in the original video also benefited cartoonisation.

6 Conclusion

In this project, we have developed AnimeGANplus, based on AnimeGAN, to perform movie-to-cartoon translation, viewed as a frame by frame instantiation of the photogaph cartoonisation task. AnimeGANplus differs from AnimeGAN in its use of layer normalization instead of instance normalization. AnimeGANplus shows fast convergence in training and compelling quality of generated cartoon images. We have provided explanations of the AnimeGANplus loss function, introduced in the original AnimeGAN, in an attempt to understand how it works at a high level.

The presented AnimeGANplus has been thoroughly investigated empirically by using three cartoon transfer styles and generating a large variety of animated images. We have used the Katna [10] key frame extraction technique to select representative frames from the input movies in qualitative evaluation of outputs. Finally, the AnimeGANplus model has also been used for cartoonisation of three videos clips, demonstrating its practical utility.

We can think of our project in terms of four tasks: dataset creation (1), implementation and training of baseline models (2), optimization of baselines (3), and preparation of reports (4). Naif and Munachiso were involved in the first three tasks, Munachiso and Almat were engaged in tasks 2, 3, and 4, and Naif helped with the fourth task.

[2] `MBZUAI_paprika_sound.mp4`

[3] Please click this YouTube link

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

[2] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[3] Gang Liu Jie Chen and Xin Chenl. A novel lightweight gan for photo animation. `https://github.com/TachibanaYoshino/AnimeGAN/blob/master/doc/Chen2020_Chapter_AnimeGAN.pdf`, 2019.

[4] Y. Chen, Y. Lai, and Y. Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9465–9474, 2018.

[5] Maciej Pesko, Adam Svystun, Pawel Andruszkiewicz, Przemyslaw Rokita, and Tomasz Trzcinski. Comixify: Transform video into a comics. *CoRR*, abs/1812.03473, 2018.

[6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[7] Levent Karacan, Zeynep Akata, Aykut Erdem, and Erkut Erdem. Learning to generate images of outdoor scenes from attributes and semantic layouts, 2016.

[8] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[9] Samet Hicsonmez, Nermin Samet, Emre Akbas, and Pinar Duygulu. Ganilla: Generative adversarial networks for image to illustration translation, 2020.

[10] Katna documentation. `https://katna.readthedocs.io/en/latest/`.

[11] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[12] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[13] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[14] Manali Shaha and Meenakshi Pawar. Transfer learning for image classification. In *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 656–660. IEEE, 2018.

[15] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. *arXiv preprint arXiv:1701.01036*, 2017.

[16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018.

[17] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.