Technological Institute of the Philippines Quezon City
College of Computer Studies
Bachelor of Science in Computer Science

Generation of Anime Characters using Generative Adversarial Networks with the Mobile Network Architecture (Mobile KanonNet)

Presented by:

Cardinoza, Christian Paul B. / BSCS
Labrador, Renzo D. / BSCS
Muralidharan, Matthew T. / BSCS
Pangan, Albert M. / BSCS
Cesista, Jeremy Reuel A. / BSCS

Submitted to:
Ms. Jasmin A. Caliwag

Section
CS33S2

June 13 2023

Table of Contents

Chapter 1

INTRODUCTION

The world of anime has captivated numerous fans who aspire to create their own original characters. However, becoming skilled at drawing requires a significant investment of time and effort, which can be a barrier for many enthusiasts. Automatic generation of anime characters presents a promising solution by providing the ability to bring customized characters to life without the need for expert artistic skills. This not only bridges the gap for aspiring artists but also facilitates the creation of custom characters without requiring professional knowledge. Moreover, the application of automatic generation can inspire character design in the animation and video game industry. Despite its potential, the creative process for generating anime characters can be challenging due to factors like writer's block and artist block, which hinder the flow of inspiration. In this research, we aim to address these challenges by developing a complex model, named 'KanonNet,' inspired by advancements in deep learning and efficient neural networks. By incorporating techniques from previous works and introducing new methods, we strive to enhance the quality, stability, and variety of generated anime characters, providing a valuable tool for artists and industry specialists alike.

Background of the Study

Numerous anime fans have considered creating their original characters. To become good at drawing, however, requires a substantial investment of time and effort; and after completing this step, then that is the time we can design our original characters.

The automatic generation of anime characters provides the ability to bring a customized character to life without the requirement for expert skill, thereby helping to bridge the gap. In addition, it allows for the creation of custom characters without the need for professional knowledge. Moreover, industry specialists can use automatic generation to inspire character design in animation and video games.

Anime has been a trend in this century and the amount of people working on it has drastically increased. Making Anime has its struggle for artists and people aspiring to draw anime-like art. Writer's Block is one of the most recent culprits in making anime art. Artists usually can't draw inspiration on making their art. Anime fans also can't find inspiration usually in drawing anime like art or even when making original characters. Evidently 31% experience creative block amongst even professionals due to deadlines or expectations and 30% experience artist block due to creative restrictions (Kulkarni, 2023).

The creative process is done by storing knowledge and recombining it to gain new information for different contexts. However, these skills of people have restrictions because learning requires a tremendous amount of time to hone skill (John, 2016; Simon, 1996). Experts are estimated to need 10,000 hours of devotion in their work to achieve breakthroughs using creativity. Attaining these levels of success are hard and time inefficient (Maata, 2023). This combination of different types of concepts is also constricted by the closeness of these concepts. Humans have an easier time to connect one information to another if they are closely related (Baddeley, 1997). These limits to the creative process can also be linked to artist block. Machine idea generation or in this case anime character generation can be a possible solution for artist block.

Anime Character Generation is a way to automatically generate anime characters by utilizing software to extract features of Anime Characters and trying to build coherent art that would look like to be drawn like an actual artist. Of this attempt to make Anime Character Generation, Deep Learning or in particular the Generative Adversarial Networks model or GANS have been striving towards making Anime Characters from random noise vectors and have been very good at that. Examples such as Rezoolab(Rezoolab, 2016) have first dealt with the topic using Vanilla GANS. Strives to make this model architecture more complex thus generating features have been seen such as trying to apply the Conditional GAN architecture towards Anime Datasets seen in the works of Hiroshiba (Hiroshiba, 2016).

The problem with these Deep Learning models and by extension Generative Adversarial Networks is that they have costly computations. The main heuristic is to make models deeper and use complex operations to achieve higher metric scores for different fields. However, in different environments that have limited computation power this should not be the case such as when running from a phone or generating images in large batches. There have been advancements to make deep learning more efficient and use less resources (Bengio, 2009; Krizhevsky et al., 2017). One of the prominent efficient neural networks is the MobileNets which is composed of separable convolutions and pointwise convolutions initially seen in XceptionNet (Chollet, 2017). MobileNets are seen as a way to factorize the convolution operation and decrease the number of parameters thus decreasing the number of computations. MobileNet can still be relatively accurate compared to InceptionNet even with decreased capacity (Howard et al., 2017).

In this research, the researchers will try to make a complex mode using the basis from works such as SRResnet (Ledig, 2016) and Spectral Normalization GANS (Aberman et al, 2019) since these methods have been proven to increase quality and stability in training Generative Adversarial Networks. New methods such as continuous upscaling and downscaling have been made similar to works in MobileNets (Miyato, 2018) where Bottlenecks have been introduced to extract low level features on a local scale and then immediately down-scaled. Other methods such as R1 Grad Penalty have been applied to models to further encourage variety in Generated Samples. The researchers dub our model named ‘KanonNet’ to pay homage towards the meaning of Kanon in Japanese which can mean “beautiful”.

Objectives of the Study

The study aims to determine and explore the possibility of using GANs in conjunction with the MobileNet architecture by Google to generate anime characters in a manner that requires less computation power than the standard GAN model. Specifically, this study aims:

- To generate anime characters in a consistent and timely manner that adheres to the general notion of what a character is.

- To extract MobileNets model using depth wise separable and compare it to the model without MobileNets.
- To evaluate the performance of GAN with or without Mobile Net using different metrics such as a five-point Likert Scale ISO/IEC 25010 according to its functional suitability, performance efficiency, and satisfaction with additional metrics such as MSE and InceptionScore.

Significance of the Study

The design and implementation of this study will significantly benefit the following:

- **Future AI developers** - The system is also beneficial for the future AI developers as it can be used as a reference tool for those developers who may want to pursue developing further this application.

Those that may require assistance in the image restoration process for their endeavors, such as:

- **Artists** - The system is beneficial for the artists as the program generates characters based on the specification provided by them. It helps artists to illustrate and create their anime characters' appearance.
- **Beginners** - The system is beneficial for enthusiasts that have no prior experience to drawing and may help them understand how drawings work by using the interpolation method. Applying different colors to a specific design can also be learned through the system.
- **Information and Communication Technology (ICT)** - The system is beneficial for demonstrating how advanced technologies like deep learning and GANs can

revolutionize the creative process, enabling the automatic generation of customized anime characters. Furthermore, by incorporating efficient neural network techniques, this research addresses the computational challenges associated with deep learning models, making automatic anime character generation more accessible and applicable in resource-constrained ICT environments.

Scope and Delimitations

The focus of this study is to apply GAN with Mobile Network architecture, GANs are well known models that learn a distribution and approximate that distribution well. The Mobile Network architecture makes computation more sustainable.

Scope

- The proponents will apply GAN with and without Depthwise Separable Convolutions.
- The dataset used are the researchers own scraped images(in inspiration to (Y. Jin, 2017)).
- The architecture is theoretically applicable to other forms of Image Generation but this study will only limit itself to Generation of Anime Characters.
- The Anime Character Generation can be a customized process and users can select different eye types and different hair types from a checkbox in the system.
- The effective pixel sizes of the model will be from 128 by 128 to 256 by 256.
- Interpolation will be done as a process to scan the distribution space of the model

and check the correlation of its learned features.

Delimitations

- The Color Space of the inputted images expected are that of the RGB space only
- The image space is images that come from professional drawings from published games and series.
- Generated images might be similar to existing work but are not entirely similar to other works.
- The dataset might be biased towards certain character types and stereotypes.

Chapter 2

THEORETICAL FRAMEWORK

Review of Related Literature

The study entitled "Generative Adversarial Networks (GAN)" has been proposed by (Goodfellow et. al., 2014) which was trying to make deep learning models be able to learn the underlying distribution of data. The original GAN conjecture was to build two machine learning models which the objective of the first(discriminator) was to identify which was real from fake and the objective of the second one(generator) was to trick the first model into believing it's generated samples. More formally it can be said that the discriminator minimizes the following loss function. $V(D,G) = E_{x \sim pdata}(x)[\log D(x)] + E_{z \sim pz}(z)[\log(1 - D(G(z)))]$ while the generator maximizes the same loss function only through the second term. This trick is done by letting gradients from the error values on the generator part of the discriminator pass through the generator through back propagation.

Other methods have been added to make GANs more stable such as introducing Spectral Normalization or Gradient Penalties according to the study entitled "Which Training Methods for GANs do actually Converge?" by (Mescheder, 2018) to inhibit the growth of the discriminator to make GAN training more stable. GANs have shown promising results on novel image generation, super resolution, denoising of images based from the study entitled "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Network" (Radford, 2015), and image transfer entitled "Unpaired Image-to-Image translation using Cycle-Consistent Adversarial Network" (Zhu et al.

2020). According to the studies entitled “Wasserstein GAN” (Arjovsky et al., 2017) and “Least Squares Generative Adversarial Network” (Mao et al., 2017), they emphasize that in general GAN training is very unstable this is due to the loss function being equivalent to minimizing the JS Divergence of the probability distributions of real images to that of fake ones, since fake samples and real samples have disconnected Manifolds at the start of training. Many methods have been introducing new loss functions such as which modify the discriminator loss to make it a distance metric rather than the original loss which has been proven to be unstable.

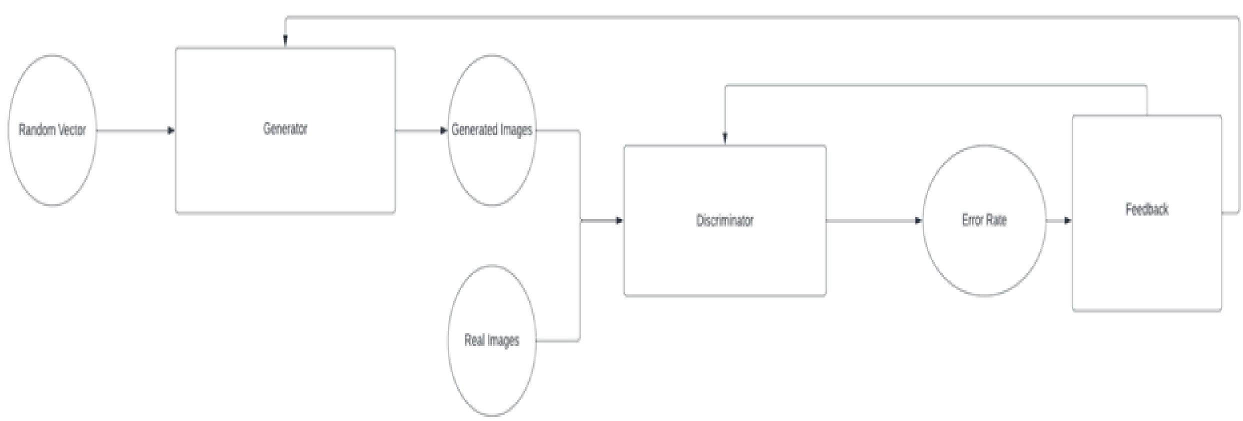
Other methods have been doing some form of regularization to ensure that the discriminator wouldn’t overpower the generator at earlier stages such as PG-GAN (Karras et al.,2018) where the generator output is constrained to some values. More GAN variants have been introduced which try to make use of more meaningful inputs rather than ones just sampled from a random distribution. Like the study entitled “Conditional Image Synthesis With Auxiliary Classifier GANs” or ACGANs by Odena et al. (2017) where GAN outputs are being controlled by a vector encoded with the intended class plus a modification of the original GAN loss function where both models try to minimize the Class Loss. Other methods such as StyleGan (Karras et al., 2018) also try to generate Novel Images in a controlled manner. Moreover, A study entitled “Style-Guided Generative Adversarial Networks for Unsupervised Anime Face Generation.” Li et al. (2022) uses a method called double-branch discriminator which aims to accurately improve anime facial structure by simultaneously shifting color/texture style. The study entitled “Using Generative Adversarial Networks for Conditional Creation of Anime Posters.” by Sankalpa et al (2022) GANS accompanied with BERT-tokenized binary genre-tags of light or heavy-hearted categorization also works to generate anime but focuses on posters. The study shows that the model was able to attain 90.17 quantitative scores of FID (Fréchet inception distance, a metric used to assess the quality of images created by a generative model).

Process Flow Diagram

Figure 1 shows the Diagram depicts the standard training procedure for a Generative Adversarial Network. Generated images are made by sampling from a random noise vector in a normal distribution then these images are fed into the discriminator and the discriminator tries to separate real images and generated images by some error rate defined by some evaluation metric. Models learn through a gradient feedback system or backpropagation (Rumelhart, 1986) .

Figure 1

Process Flow diagram of Training Procedure of GAN

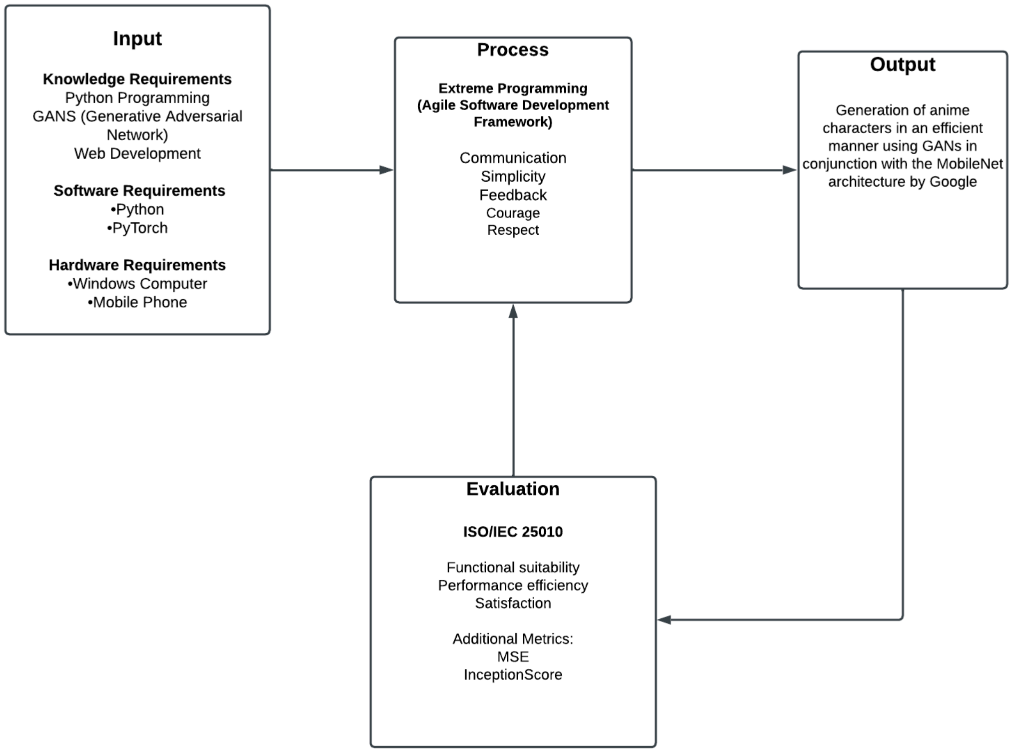


Design Concept

Figure 2 shows the Conceptual Framework follows the input-process-output model, which includes the knowledge, software and hardware requirements, software development life cycle model for the process, and the system for the output. In the software development life cycle, various methodologies can be adopted, such as Extreme Programming (XP), which is known for its emphasis on effective teamwork and high-quality software development. XP values communication, simplicity, feedback, courage, and respect as its core principles. These values drive the XP process, where constant communication among team members promotes collaboration and shared understanding. Simplicity ensures that the software solution is kept as straightforward and maintainable as possible. Feedback loops enable continuous improvement by incorporating user input and responding to changing requirements. Courage encourages developers to take risks and experiment with new ideas, while respect fosters a positive and supportive team environment(McDonald, 2023b). When evaluating the software system, Information Technology and computer science computing individuals can use ISO 25010 standards and other metrics like MSE and Inception Score in alignment with the XP principles to assess its quality and performance.

Figure 2

Conceptual Framework of the KanonNet

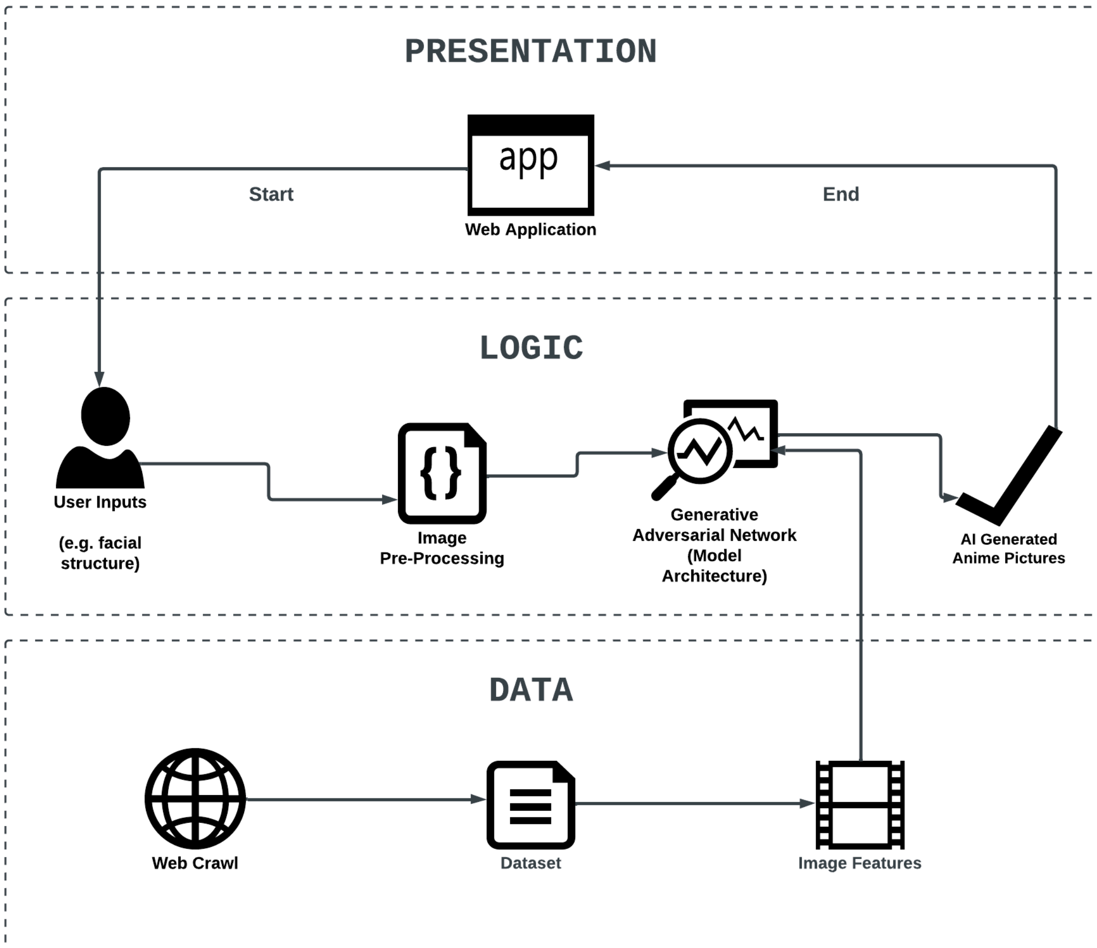


System Architecture

Figure 3 shows the System Architecture that encompasses three tiers: Presentation, Logic, and Data. In the Presentation and Logic tiers, the application requires user inputs related to the desired character's facial structure, such as hair or eye color, as well as the preferred art style. These inputs undergo preprocessing and are utilized in conjunction with Generative Adversarial Networks (GANs) and the MobileNet architecture developed by Google. The outcome of this process is the generation of anime pictures that fulfill the user's specifications. For the data tier of the architecture involves training the model using a dataset specifically acquired for this purpose. This dataset comprises numerous examples of anime facial structures, serving as a reference for the AI-generated anime images.

Figure 3

System Architecture of the KanonNet Web Application



Definition of Terms

Term	Definition
GANS	Generative Adversarial Networks, a neural network

	architecture focused on translating some random noise to a distribution close to that of real data. Many GAN variants have been introduced throughout its time all focusing on making GAN Training stable or generating better quality output.
Noise Vector	a random vector that all values have been sampled on some random probability distribution such as Gaussian Distribution and Uniform Distribution.
Neural Network	a machine learning Model where data is fed in, and it tries to minimize some error function. Usually used when available data is large since Neural Networks have been shown to learn better and better with more data available.
Error Function	an evaluation metric that uses the model for learning and for humans to evaluate. Error functions are differentiable, have a local optimal, between infinity and 0.
Back Propagation	Method used to get the respective error of the underlying model parameters to be able to learn the corresponding weight adjustments to minimize the error function.

Chapter 3

RESEARCH METHODOLOGY

This chapter discusses the methodology that is used in the study. The proponents present the research methods, data collection, software methods, and the background of the algorithms.

Project Materials

A variety of programs and libraries will be used to create the web application. This section outlines all of the components, including the specification of the hardware, and the various softwares utilized in this study

Software Requirements

Table 1

Front End Development

Bootstrap	open-source framework that simplifies web development by providing pre-built CSS styles, JavaScript components, and a responsive grid system. It enables developers to create visually appealing, responsive websites and applications quickly and efficiently.

HTML	provides the foundation for web development and is essential for creating the structure and layout of webpages, while also enabling the incorporation of other technologies like CSS and JavaScript to enhance the functionality and presentation of websites and web applications
CSS	includes aspects such as layout, colors, fonts, spacing, and more for easier management and consistency across multiple webpages, providing greater control over the design and appearance of websites and web applications.
Visual Studio Code	is an Integrated Development Environment(IDE) developed by Microsoft. It will be used in the development of the program.
Google Colab	provides a Jupyter notebook environment, allowing you to write and execute Python code in cells. The notebook interface is user-friendly and facilitates code organization, documentation, and collaboration.

Back End Development

Python	A high-level and general purpose programming language that emphasizes code readability and supports a wide range of libraries.

Flask	is an open source python library that provides interfaces for TensorFlow libraries and artificial neural networks.
Pytorch	open-source machine learning framework that provides a wide range of tools and functions for building and training neural networks. PyTorch supports both CPU and GPU computation, making it suitable for various hardware configurations.

Hardware Requirements for training and web development

Desktop / Laptop. The table 2 shows the operating system for the system required for the training of Generative Adversarial Network and web application development is Windows 11 as it is the development OS. Although the application is built with cross-platform libraries. Processor can be the minimum standard of the modern web for access, but for development, any modern processor that has at least 4 cores and 4 threads is required. The Random Access Memory is set as well to the minimum standard of the web for access, and 8GB and above for training and web development. The GPU used for training must be at least gtx 1050ti to properly train the Generative Adversarial Network and for web development.

Table 2

Specification of Desktop / Laptop

OS	Windows 11
CPU	ryzen 5/i3

RAM	8GB RAM
GPU	gtx 1050ti

Hardware Requirements for running the web application

The table 3 of this research project entails the utilization of devices that possess internet connectivity and were manufactured no earlier than five years ago. The devices should have the capability to access and connect to the internet. Please ensure that the devices meet the following specifications:

Table 3

Specifications

Internet Connectivity	devices must have the ability to connect to the internet. This can be achieved through Wi-Fi, Ethernet, or any other means of internet connectivity.
Age of Devices	devices should have been manufactured within the last five years. This ensures that they possess the necessary hardware capabilities and are compatible with modern software and internet technologies.
Compatibility	devices should be compatible with the software or

	applications required for the research project. Please verify that the devices can support the required software versions and meet any additional compatibility requirements.
Sufficient Processing Power	devices should have adequate processing power to handle the research tasks and the associated internet-related activities. This includes running any necessary software, accessing online resources, and performing data analysis or simulations, depending on the nature of the project.
Sustained Internet Access	devices have reliable and sustained internet access throughout the research project. This ensures that data can be collected, transmitted, and shared seamlessly over the internet.

Project Design

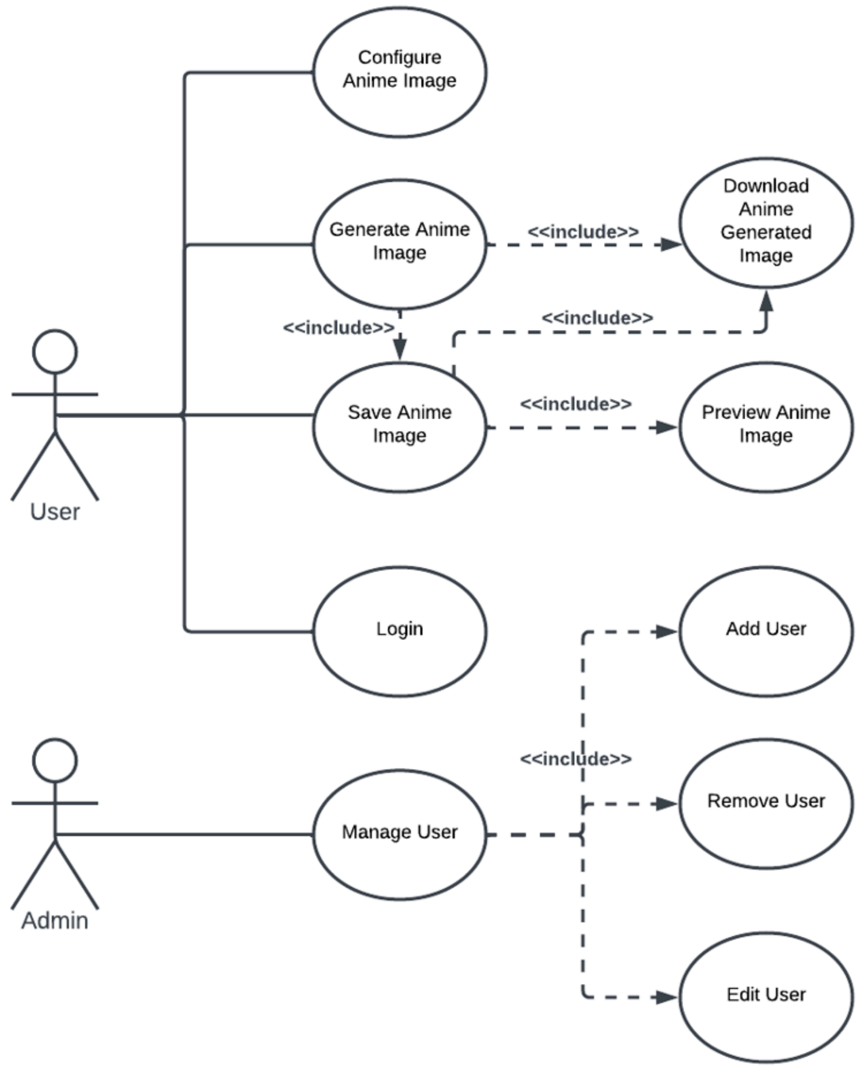
The project design of our research is illustrated through a collection of diagrams that depict the essential procedures and processes. Within this section, an exploration of diverse diagrams, including the use-case diagram, context diagram, and sample screen designs, is presented. These diagrams serve as the foundation for the development of our project.

Use Case Diagram

Figure 4 illustrates the use case diagram of the system. In the use case diagram, the application consists of two primary actors: the administrator and the user of the system. The administrators are primarily responsible for managing the user list. This includes removing the user, adding users and modifying the users details as well. On the other hand, the user is the one who will primarily utilize the system. Users have the capability to configure anime images by selecting models, a type of model, hair and eye color, style, and noise. The application will generate an image of an anime based on the configuration of the users. In addition, users possess the functionality to save the generated images within their profiles and have the option to download them at their convenience.

Figure 4

Use Case Diagram of KanonNet Web and Mobile Application

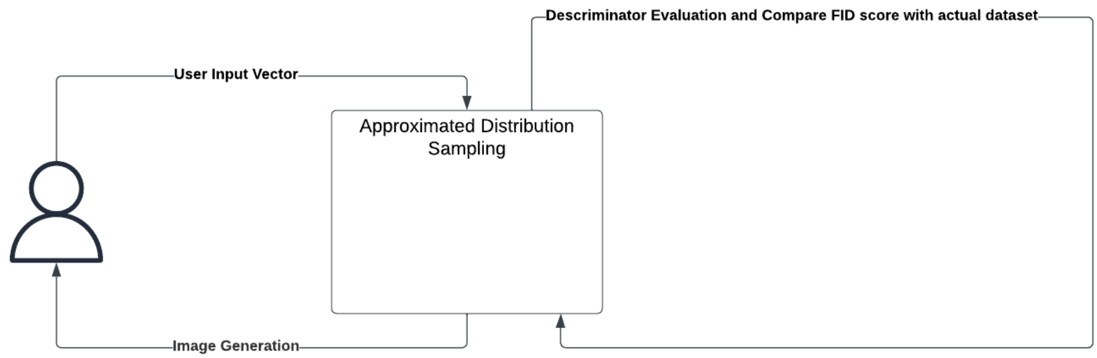


Context Diagram

Figure 5 illustrates the interaction between the user and the system. The user provides a vector as input to the system. Subsequently, the system processes the provided vector and generates a corresponding character. Additionally, the system employs a discriminator evaluation, comparing the FID score to the actual dataset, in order to enhance the quality of image generation.

Figure 5

Context Diagram of KanonNet



Sample Screens

Figures 6, 7 and 8 consist of Sample screens like the landing page, main page, and log-in page, designed to facilitate a seamless and efficient use and research experience.

Figure 6

Sample of landing page

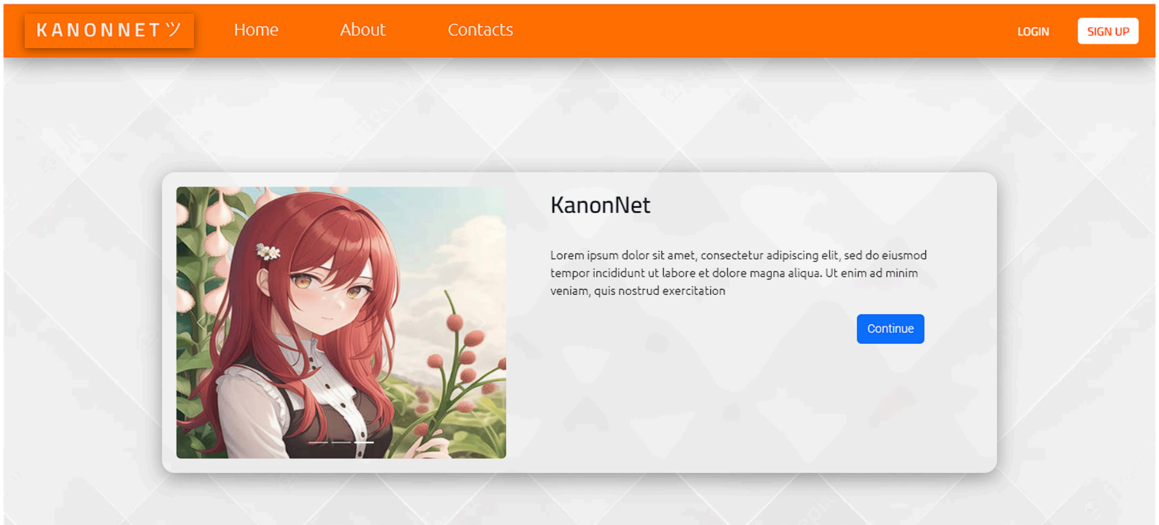


Figure 7

Sample of login page

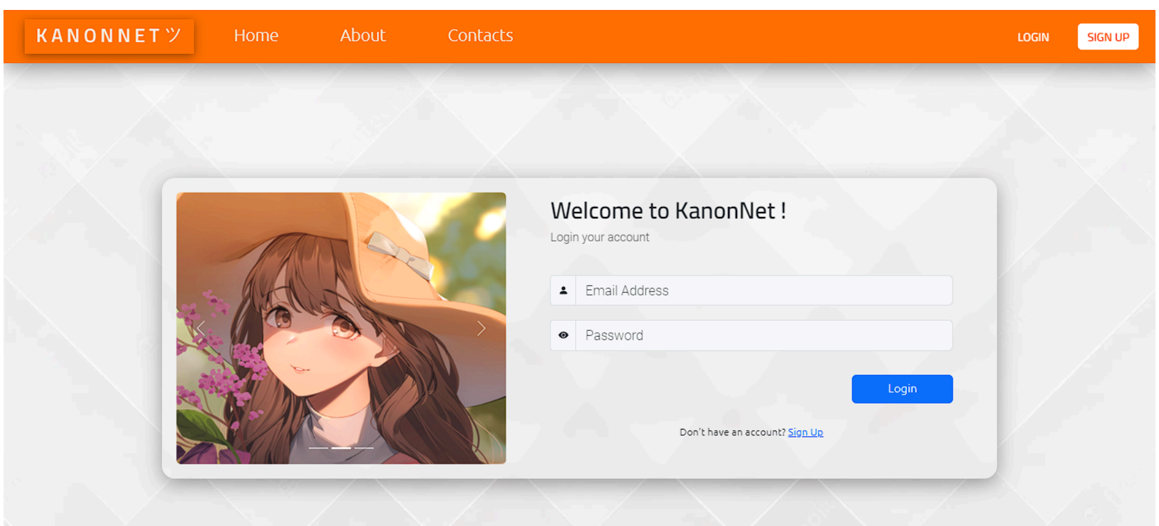
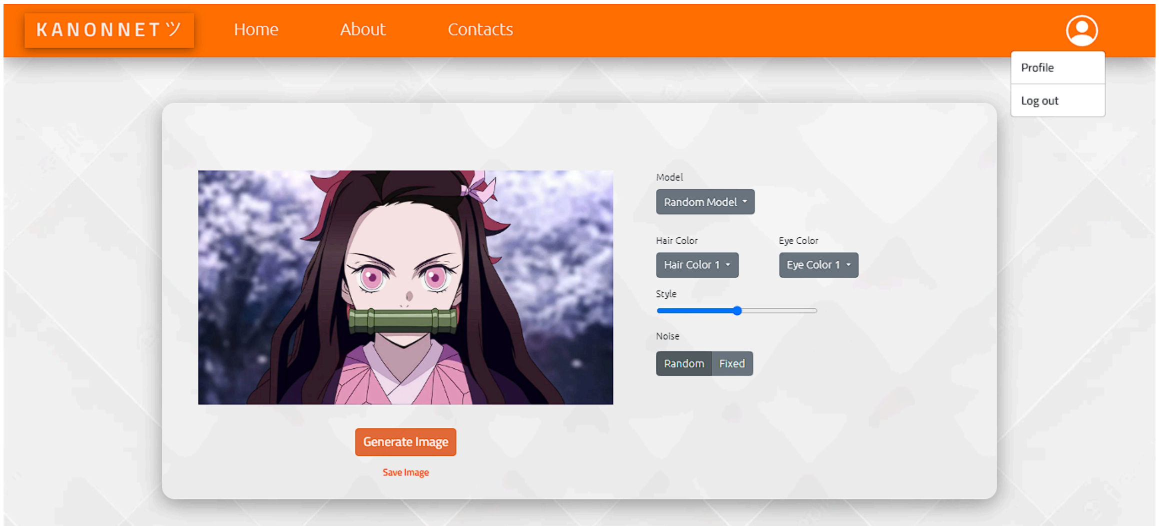


Figure 8

Sample of home page



Project Development

Data Collection

Quality of generated images is dependent on the quality of the dataset as highlighted in the paper of Yanghua(Jin, 2017). Images Gathering is the same with Yanghua's Paper. All images are gathered from getchu by first running a SQL query on ErogameScape as highlighted in the Appendix. Images are then downloaded by using a web scraping script on the gathered image urls. Figure 9 shows the Using of lbpcascade(Ultraist, 2014) to crop out anime face images from the full body picture.

Figure 9

Samples of lbcapscade face cropping

Images are then cleaned by first removing all of the images less than 100 pixels and then manual cleaning after.

Model Training

Images are first loaded using the Pytorch Dataloader resized to 128 by 128 and values are scaled from [0,255] to [-1,1] for training. Models are run on Google Colab for training with the underlying methods and neural architecture stated on the Project Design. A Nvidia P-100 GPU is used at a runtime of 12 hours to be able to train the models, however it is observed a Nvidia 1050ti may suffice given a small batch size. A Learning Rate Scheduler(Howard, 2017) is used on the Model with a gamma rate of 0.875. Models are evaluated on a 10 epoch basis and compared with other epochs based on human perceived quality.

Figure 10

Samples of generated images in epoch 30 of Training of KanonNet

Website

All trained models are exported, it has been observed that after the 30th epoch images seem to have downgraded quality. The 30th epoch of KanonNet is used when sampling images. The website has a feature of selecting the starting image and ending image to be able to use interpolation for users to see the interpolation between two images in an animation. Interpolation lets people see how the model sees the distribution of anime characters. Other techniques the website offers is to provide 4 data points the user may control to have some control over the generation of images of the model. Another method seen in BigGans is the truncation technique which allows for more consistent images, Observations seem to point a value between 1 and 1.5 in truncation norm std and 0 in the mean allows the model to have consistent image quality.

Figure 11

Samples of generated images in epoch 40 of Training of KanonNet. Observable Quality Loss.

Algorithm Implementation

How does the algorithm work?

Figure 12 of the Generator Architecture draws inspiration from the SRResnet GAN architecture which also uses b residual blocks first, however there are slight modifications to this method in this generator. It still retains the first b residual blocks however the pixel shuffle blocks now have a block before it called the Upscaling block, it draws inspiration from the works in MobileNetV2(Radford, 2015) where inverted residual blocks increase the number of channels and then downscale the number of channels. Upscaling blocks are made to increase the number of potential features that the model has to have a wide variance in generated characters, however, skip connections are added to ensure that there is no possible information loss in upscaling and downscaling. Adding the Pixel Shuffle layer in combination with these bottlenecks of information allows for less information loss than the ones described in Mobile Nets. In summary the model comprises an initial 16 residual blocks and 4 modified sub-pixel blocks with an added upscaling block.

Figure 12

Generator Architecture

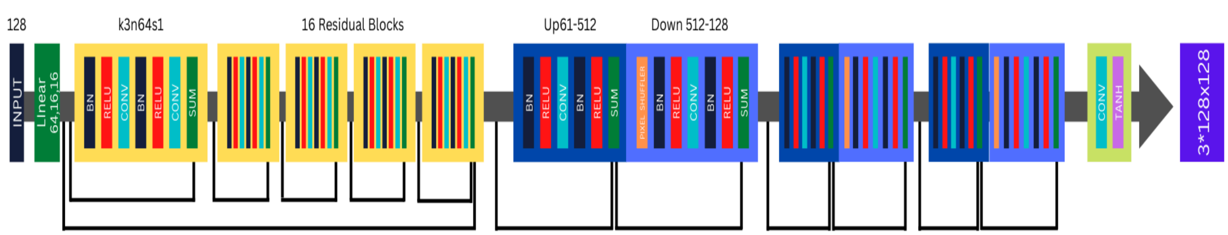
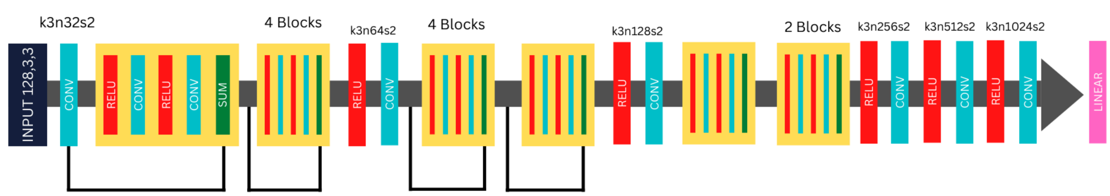


Figure 13 The Discriminator Architecture uses a standard ResNet architecture but strided convolution cuts off residual connections and restarts since a 1x1 convolution with a stride of 2 would produce information loss and may hurt the quality of generated images.

In addition, all weights of the generator and discriminator are initialized with a mean of 0 and a standard deviation of 0.02. Both models are trained on the ADAM optimizer and both beta2 are set to 0.9 while beta1 is set to 0.

Figure 13

Discriminator Architecture



Improved GAN Training

Here are all the methods that KanonNet uses to stabilize GAN training. Spectral Normalization is both used in the Generator and Discriminator as highlighted in Self-Attention GANS it helps during training to also use Spectral Normalization on the Generator(Zhang, 2018). Using Hinge-Loss(Brock, 2018) as seen on BIGGAN on the Discriminator outputs helps during training at it provides a smooth derivation helping the generator during the early phases of training and applying an Exponential Moving Average(Yazici, 2018) to the Generator weights helps increase the quality of generated images of the model without affecting the optimization of GANS.

Hinge Loss:

$L(y) = \max(0, 1\text{-}D(G(z))$ for generated samples or $\max(0,D(x))$ for real samples.

Project Evaluation

The project evaluation of KanonNet will focus on the performance of GAN with or without Mobile Net using different metrics such as a five-point Likert Scale ISO/IEC 25010 with additional metrics such as MSE and InceptionScore.

ISO/IEC 25010

The project's ISO standard for the study was ISO/IEC 25010. Functional stability, performance efficiency, compatibility, usability, dependability, security, maintainability, and portability are the eight characteristics that make up ISO/IEC 25010. The characteristics that will be used for this study are according to its functional suitability, performance efficiency, and satisfaction.

Respondents

The respondents are composed of CS/IT undergraduate students and computing professionals. The undergraduate students will be able to evaluate the outer appearance of the system. While the computing professionals can evaluate the system thoroughly and give pointers to further improve the system.

Multiple Constraints and Tradeoff Analysis (Sensitivity Analysis)

This section outlines the algorithm and discusses the constraints utilized to decide the algorithm. This section also provides the design trade-offs, trade-offs analysis and sensitivity analysis.

Constraints Definition

In the proposed application model, the primary algorithm that serves as the best appropriate algorithm for this model is the FID (Frechet Inception Distance). FID is usually used to compute the distance between feature vectors calculated from real and generated images. In the application, in order to calculate the FID score, a pre-trained CNN (Inception model) was used to extract vision-relevant features from both real and fake samples. To accurately measure visual similarities between generated anime-style illustrations and real images, using the Inception model trained on ImageNet is not appropriate. This is because the ImageNet dataset does not include anime-style images. Instead, it is recommended to replace the Inception model with an Illustration2vec feature extractor model. This model is specifically designed for extracting features from illustrations and can provide more reliable measurements for evaluating the visual similarities between generated and real anime-style images. The main idea to evaluate the FID score for the application, approximately 12000+ images must be retrieved from the real dataset. A fake sample must also take into account to generate a corresponding fake sample by employing the relevant conditions for each real image. Both the real and fake images are then passed through the Illustation2vec feature extractor, resulting in 4096-dimensional feature vectors for each image. Hence, the FID score is calculated by comparing the collection of feature vectors obtained from the real samples with those obtained from the fake samples.

Figure 14

FID Formula

$$d^2((\boldsymbol{m}, \boldsymbol{C}), (\boldsymbol{m}_w, \boldsymbol{C}_w)) = \|\boldsymbol{m} - \boldsymbol{m}_w\|_2^2 + \mathrm{Tr}\big(\boldsymbol{C} + \boldsymbol{C}_w - 2\big(\boldsymbol{C}\boldsymbol{C}_w\big)^{1/2}\big)$$

where:

d^2 = Distance Score

$\boldsymbol{m} - \boldsymbol{m}_w$ = Feature-wise mean of the real and generated images

$\boldsymbol{C}\boldsymbol{C}_w$ = Covariance matrix for the real and generated feature vectors, often referred to as sigma.

Trade-offs

Training Time vs. Model Performance

A trade-off exists between training time and model performance in our thesis project. While longer training times generally lead to improved model performance, there is a practical limit to the time available for training. Striking a balance is crucial to ensure practicality in achieving optimal results. Extending the training time beyond a certain point may yield diminishing returns or may not be feasible due to project constraints, such as limited computational resources or time restrictions.

Therefore, it is necessary to carefully determine the optimal training duration that maximizes model performance within the given constraints and resources.

Spectral Normalization vs. Training Speed

A trade-off exists between spectral normalization and training speed in our thesis project. Spectral normalization is beneficial for stabilizing the training process of Generative Adversarial Networks (GANs). However, the introduction of spectral normalization may increase the computational overhead and slow down the training speed. It is essential to strike a balance between achieving training stability through spectral normalization and maintaining an acceptable training speed. The optimal approach involves finding the right configuration and parameter settings that provide a good trade-off between stability and training efficiency, considering the available computational resources and project time constraints.

Hinge Loss vs. Discriminator Performance

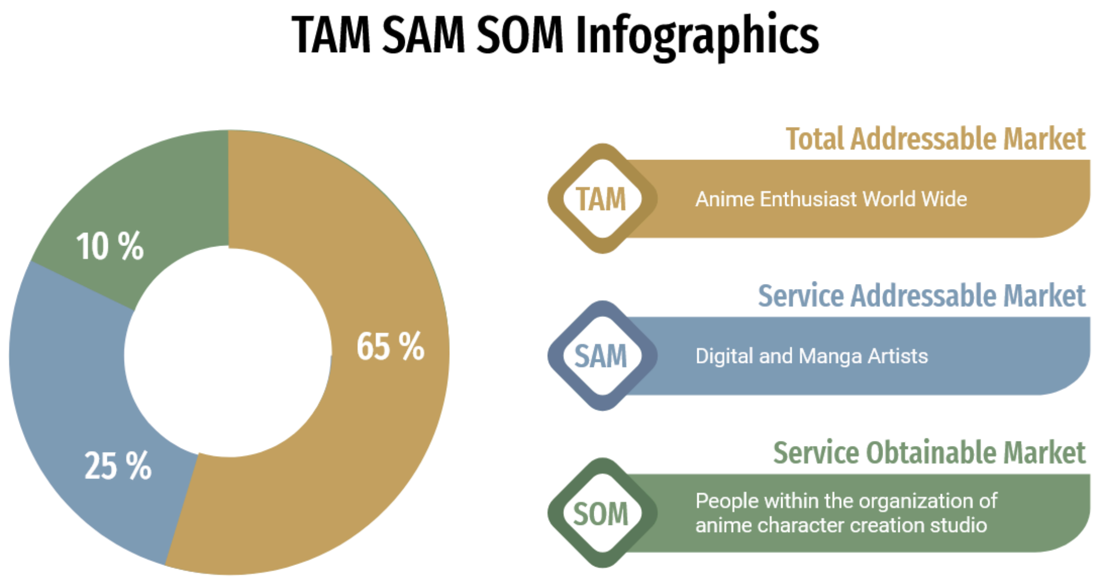
a trade-off exists between the use of hinge loss and the performance of the discriminator. Hinge loss, particularly the max(0, 1-D(G(z))) formulation for generated samples, aids in training by offering a smooth derivative and assisting the generator in the early stages. However, employing hinge loss may affect the discriminator's ability to accurately distinguish between real and generated samples. This trade-off lies in finding the right balance between optimizing the generator's performance and maintaining the discriminator's accuracy in differentiating real and generated samples. Careful consideration should be given to the selection and tuning of loss functions to strike a balance that maximizes the overall performance and training stability of the GAN model

Market Model

Figure 15 shows the market model of the application. The market model comprises TAM(Total Addressable Market), SAM (Serviceable Addressable Market), SOM (Serviceable Obtainable Market). The TAM segment covers 65% of the total population which indicates the anime enthusiast world wide. According to the study entitled “20+ Anime Statistics & Facts: How Many People Watch Anime?” by (Ferjan, 2023), more than a third of the world population with a total equivalent of 2.88 billion people are categorized as anime enthusiasts. On the other hand, the SAM segment covers 25% of the total population which indicates the digital and manga artists. Recent study shows that roughly 61 million are registered as digital artists which is equivalent to over 500,000 members that are part of a large and active community of digital artists worldwide. Lastly, the SOM segment covers 10% of the total population which indicates individuals that are part of the organization within the anime character creation studio. Another study shows that approximately 8,076 full-time employees are working in the anime industry in Japan in 2018.

Figure 15

Market Model of the KanonNet

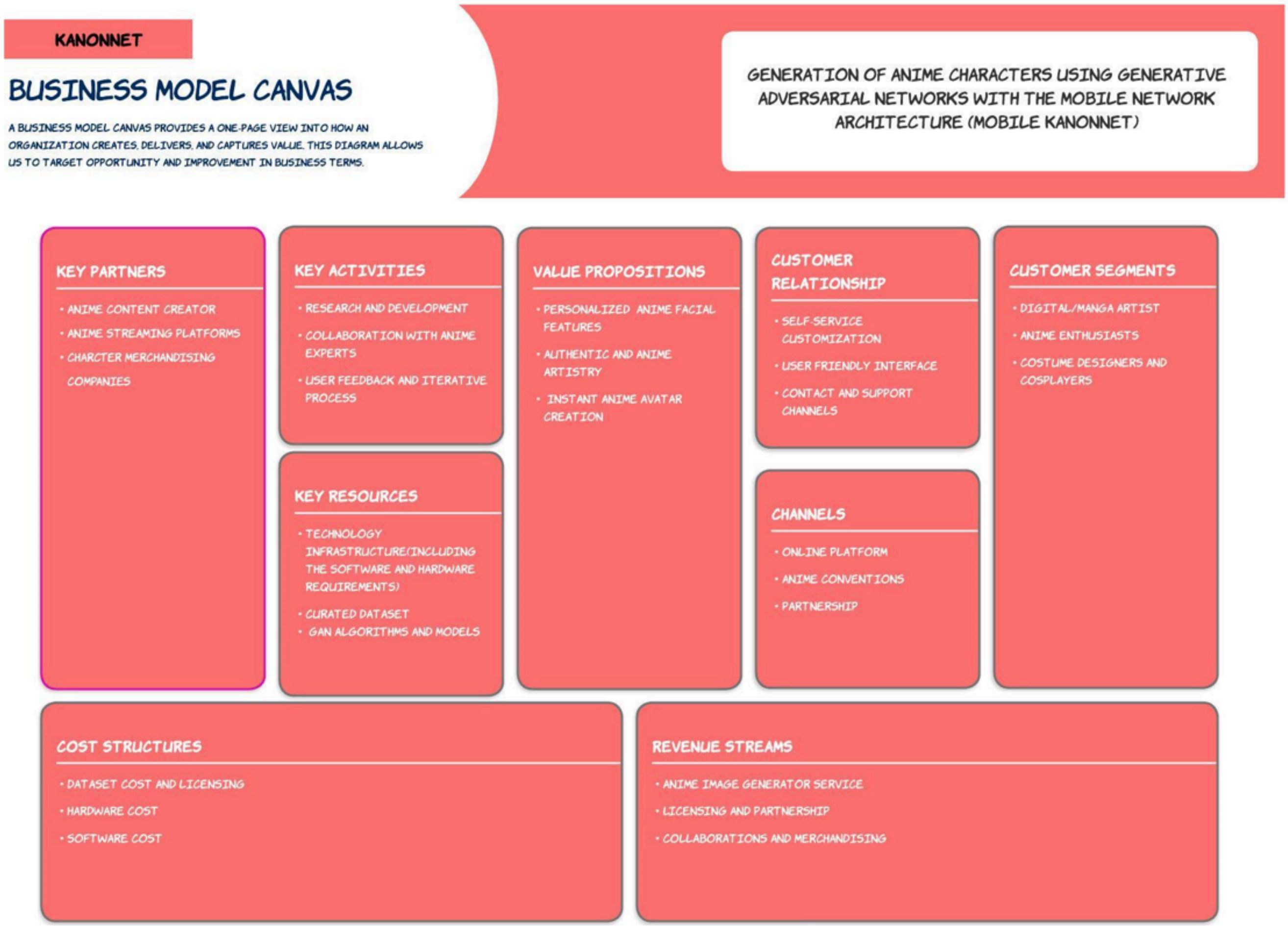


Business Model

Figure 16 shows the business model that consists of Key Partners, Key Activities, Key Resources, Value Propositions of the Team and Product, Customer Relationships, Project Promotion Channels, Customer Segments, Cost Structure, and Revenue Streams.

Figure 16

Business Model of KanonNet


KANONNET
BUSINESS MODEL CANVAS
A BUSINESS MODEL CANVAS PROVIDES A ONE-PAGE VIEW INTO HOW AN ORGANIZATION CREATES, DELIVERS, AND CAPTURES VALUE. THIS DIAGRAM ALLOWS US TO TARGET OPPORTUNITY AND IMPROVEMENT IN BUSINESS TERMS.
GENERATION OF ANIME CHARACTERS USING GENERATIVE ADVERSARIAL NETWORKS WITH THE MOBILE NETWORK ARCHITECTURE (MOBILE KANONNET)
KEY PARTNERS
• ANIME CONTENT CREATOR
• ANIME STREAMING PLATFORMS
• CHARCTER MERCHANDISING COMPANIES
KEY ACTIVITIES
• RESEARCH AND DEVELOPMENT
• COLLABORATION WITH ANIME EXPERTS
• USER FEEDBACK AND ITERATIVE PROCESS
KEY RESOURCES
• TECHNOLOGY INFRASTRUCTURE(INCLUDING THE SOFTWARE AND HARDWARE REQUIREMENTS)
• CURATED DATASET
• GAN ALGORITHMS AND MODELS
VALUE PROPOSITIONS
• PERSONALIZED ANIME FACIAL FEATURES
• AUTHENTIC AND ANIME ARTISTRY
• INSTANT ANIME AVATAR CREATION
CUSTOMER RELATIONSHIP
• SELF SERVICE CUSTOMIZATION
• USER FRIENDLY INTERFACE
• CONTACT AND SUPPORT CHANNELS
CHANNELS
• ONLINE PLATFORM
• ANIME CONVENTIONS
• PARTNERSHIP
CUSTOMER SEGMENTS
• DIGITAL/MANGA ARTIST
• ANIME ENTHUSIASTS
• COSTUME DESIGNERS AND COSPLAYERS
COST STRUCTURES
• DATASET COST AND LICENSING
• HARDWARE COST
• SOFTWARE COST
REVENUE STREAMS
• ANIME IMAGE GENERATOR SERVICE
• LICENSING AND PARTNERSHIP
• COLLABORATIONS AND MERCHANDISING

Chapter 4

Results and Discussion

This chapter includes the project description, discussion of its development, including the detailed structure of the mobile application and its user interface, and the training and finetuning of the deep learning model. Review and discussion of the findings, analysis, and interpretation of the results and evaluation of the study are also covered in this chapter.

Project Description

The study was done to create a complex model based on previous works such as SRResnet and Spectral Normalization GANs, known for improving the quality and stability of GAN training.

To meet the project objectives, the model is inspired by MobileNets to incorporate methods such as continuous upscaling and downscaling, which involve extracting low-level features on a local scale and then immediately downscaling. Additionally, techniques like R1 Grad Penalty are applied to encourage variety in the generated samples.

Data Gathering

The primary dataset utilized in this study is retrieved from the Getchu website. Getchu is a website that offers information and facilitates the sale of Japanese games. On this website, there are sections dedicated to introducing characters, which include standing pictures of the characters. The characters' images are diverse enough which means they have different variations of hair color, eye color, skin color, facial structure, eye glasses or no glasses, etc. Retrieving the data from the website is a very complex process as it involves numerous steps. The following steps include: executing SQL query on ErogameScape’s Web SQL Api to retrieve the Getchu page link for every game. To download images from the gathered links, web scraping methods were used. Lastly, lbpcascade to crop out anime face images from the full body picture.

Development Discussion

Extreme Programming (XP) or Agile stand out as effective approaches that foster collaboration, consistent communication, and a culture of continuous improvement. By adopting an appropriate software development life cycle methodology, project teams can lay the groundwork for successful outcomes, ensuring that all stakeholders are actively involved, ideas are shared openly, and progress is constantly monitored and refined. Firstly, exploring alternative methods or advancements in GAN architectures, such as Progressive GANs or StyleGAN, could potentially improve the quality and diversity of generated images. Additionally, investigating novel discriminator architectures or techniques like Wasserstein GANs can contribute to more accurate discrimination between real and generated images. Evaluating the system's performance using relevant metrics, including ISO 25010 standards, MSE, and Inception Score, ensures a comprehensive assessment of software quality and performance. The system architecture's three-tier structure, encompassing Presentation, Logic, and Data tiers, provides a foundation for scalability and extensibility, allowing for the integration of additional features and accommodating future enhancements. Continuously improving the dataset by acquiring diverse examples of anime facial structures and exploring data augmentation techniques can further enhance the training process. By addressing these areas of development, the application can evolve to generate higher-quality anime pictures, improve user satisfaction, and ensure a robust and efficient development process.

Project Structure

This section will discuss how the project is created in terms of design, features, and other technical aspects. This section will also include the screenshots of the developed web application and how the algorithm is implemented from the web application to the model.

In Figure 17 and 18, our application showcases its ability to generate a new anime character effortlessly with just a single click of the generate button. The speed at which it accomplishes this feat is truly remarkable, taking less than a second to create a unique character. This achievement perfectly aligns with our project's primary objective: To generate anime characters in a consistent and timely manner that adheres to the general notion of what a character is.

Figure 17

System Generates Anime Images with Random Preference.

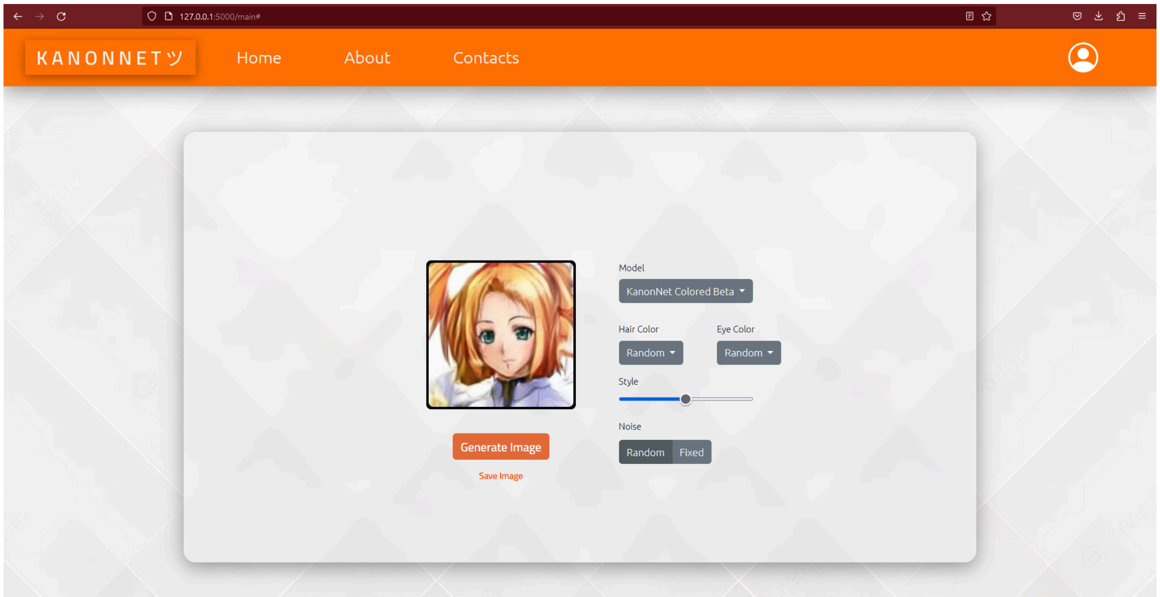
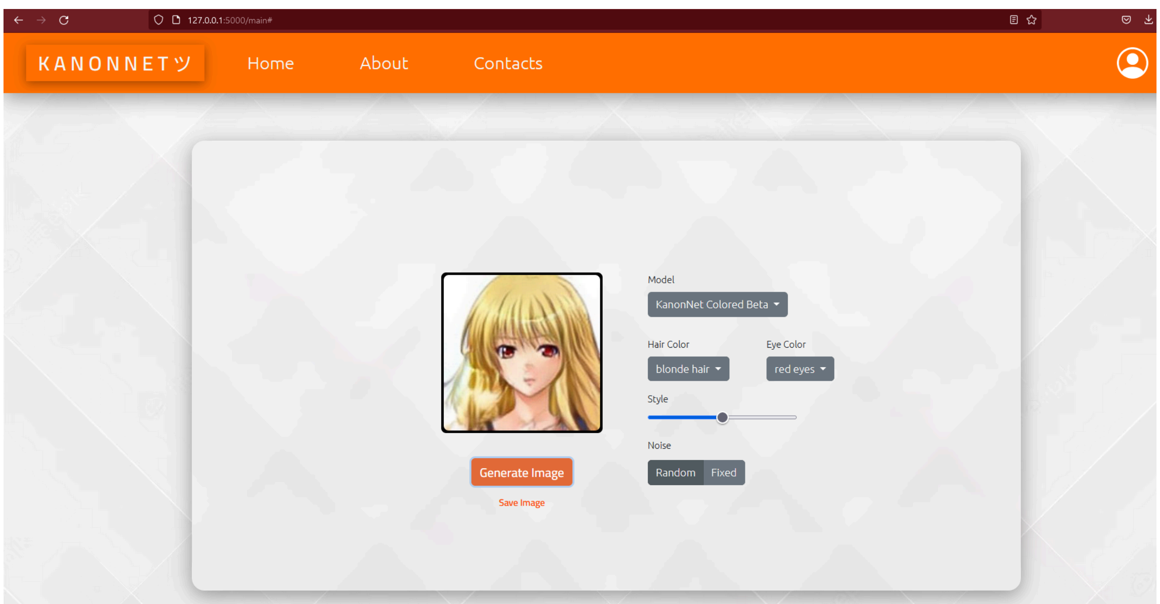


Figure 18

System Generates Anime Images with User Customization.



Figures 17 and 18 demonstrate a convenient solution for users seeking anime character generation by eliminating the need for manual drawing or extensive artistic skills. Its automated generation process streamlines the workflow, saving artists valuable time and effort. Artists can efficiently explore various character designs and iterate on visual elements like facial features, hairstyles, and eye color to achieve their desired results. This system's accessibility and efficiency have the potential to benefit artists, enthusiasts, and creators. It removes the barriers of starting from scratch, allowing for the generation of a diverse range of anime characters. This fosters creativity and exploration, enabling users to quickly and effortlessly create captivating anime facial characters. Whether for storytelling, gaming, or other creative endeavors, the system empowers users to generate compelling anime characters with ease. By offering a level of accessibility and efficiency that caters to both novice creators and experienced artists, this system opens up new possibilities in anime character generation. It simplifies the process, making it more accessible to a wider range of users and encouraging creative experimentation. The system's ability to swiftly generate anime characters addresses the challenges faced by artists who may struggle with the time-consuming task of drawing or creating characters from scratch.

Figure 19

Generated Anime Images using KanonNet Original (Default Model) without MobileNets

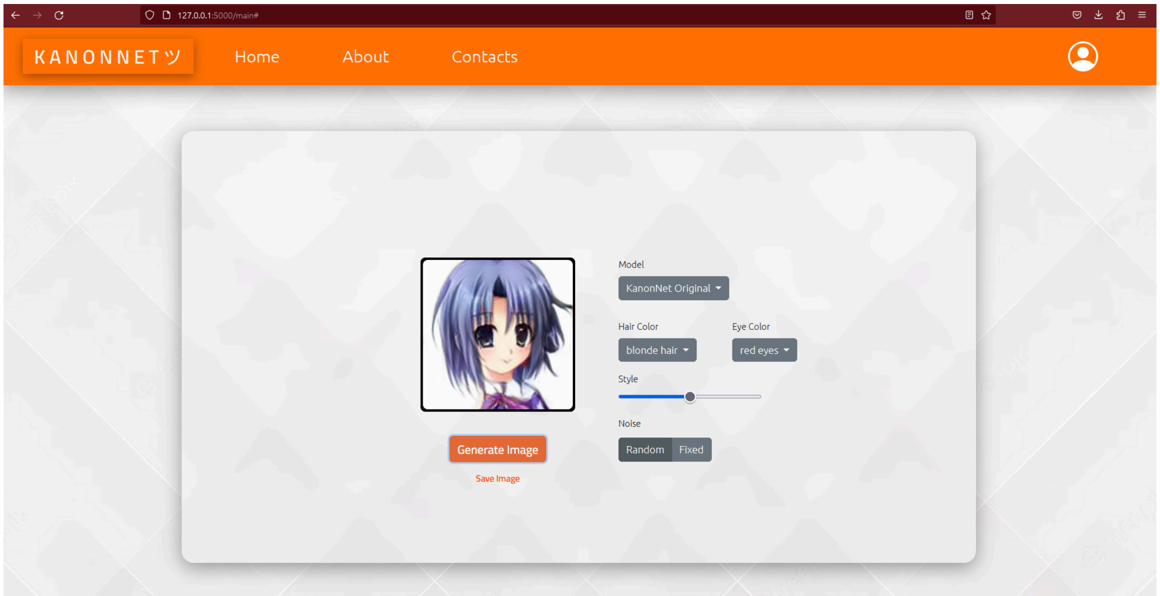
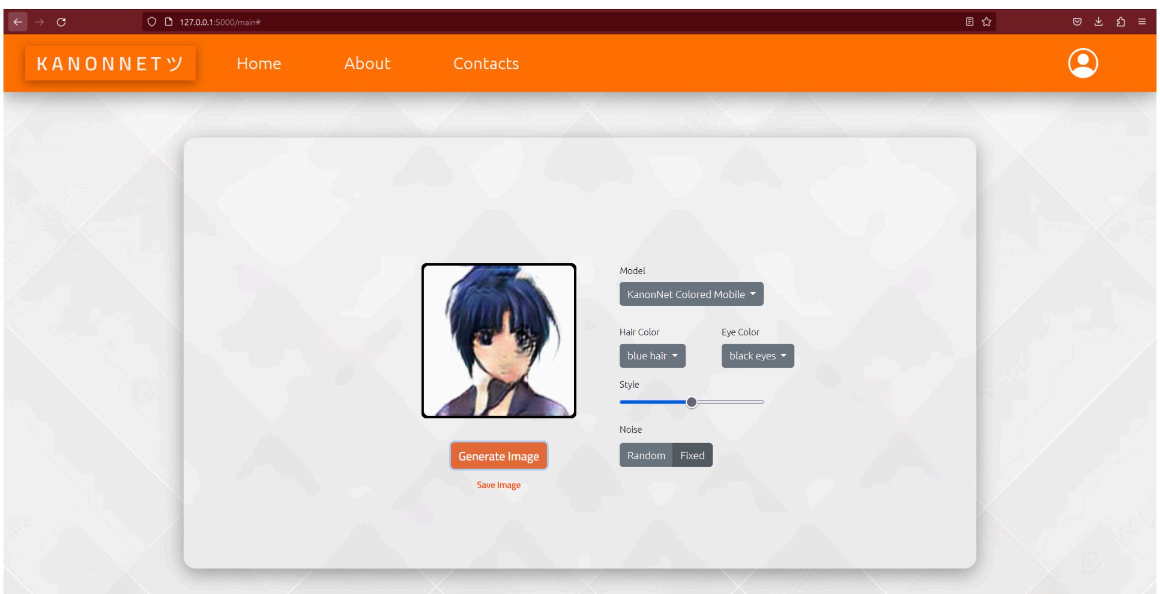


Figure 20

Generated Anime Images using KanonNet Colored Mobile with MobileNets



Figures 19 and 20 demonstrate the comparison of the generation of anime characters with KanonNet Original (Default) without MobileNets and KanonNet Colored Mobile with MobileNets.

Project Evaluation

The project will be evaluated based on several key metrics. These include the quality of the generated anime characters, the stability of the model during training, and the diversity of the generated samples. The researcher will also assess computational efficiency of the model, particularly when using the MobileNet architecture. To evaluate the quality of the generated anime characters, researchers will use both quantitative and qualitative methods. Quantitatively, researchers will use metrics such as the Inception Score and the Frechet Inception Distance, which measure the quality and diversity of the generated images. Qualitatively, researchers will conduct user studies where participants rate the realism and aesthetic appeal of the generated characters.

Table 4

Usability Criterion Based on User Respondents

Sub-characteristics	Mean Average	Interpretation
Learnability	4.7	Strongly Agree
Operability	4.5	Strongly Agree
User Interface Design	4.8	Strongly Agree
Mean Average	**4.66**	Strongly Agree

Table 5

Performance Efficiency Criterion Based on User Respondents

Sub-characteristics	Mean Average	Interpretation
Time Behavior	4.9	Strongly Agree
Capacity	4.8	Strongly Agree

Resource Utilization	4.6	Strongly Agree
Mean Score	**4.76**	Strongly Agree

Table 6

Functional Suitability Criterion Based on User Respondents

Sub-characteristics	Mean Average	Interpretation
Functional Completeness	4.7	Strongly Agree
Functional Appropriateness	4.9	Strongly Agree
Functional Correctness	4.6	Strongly Agree
Mean Score	**4.73**	Strongly Agree

Table 7

Summary Result of KanonNet Users' Respondents

Characteristics	Mean Average	Interpretation
Usability	4.66	Strongly Agree
Performance Efficiency	4.76	Strongly Agree
Functional Suitability	4.73	Strongly Agree
Mean Average	**4.72**	Strongly Agree

Computing Standards

The KanonNet establishes a comprehensive framework comprising a set of guidelines for evaluating the quality of software applications specifically designed for classification tasks employing convolutional neural networks. In pursuit of this objective, the researchers adopted the ISO 25010 software quality standard as the basis for assessing the web application's quality across three key dimensions: usability, performance efficiency, and functional suitability. The usability criterion entails an evaluation of the system application's ability to satisfy user requirements and accomplish defined objectives. By examining this criterion, researchers can ascertain whether the system application effectively fulfills its intended purposes. The performance efficiency criterion centers on evaluating the system application's capacity to deliver optimal performance, thereby ensuring a superior and dependable user experience. Additionally, it facilitates the identification and elimination of potential system bugs or glitches. The functional suitability aspect is expected to encompass precision, completeness, and compliance with the established standard. This criterion verifies whether the system functions accurately and adequately fulfills the specified requirements. It also aids researchers in identifying critical areas within the system that demand particular attention and focus.

Figure 20

Comparison of Discriminator and Generator Loss with and without Mobile Nets.

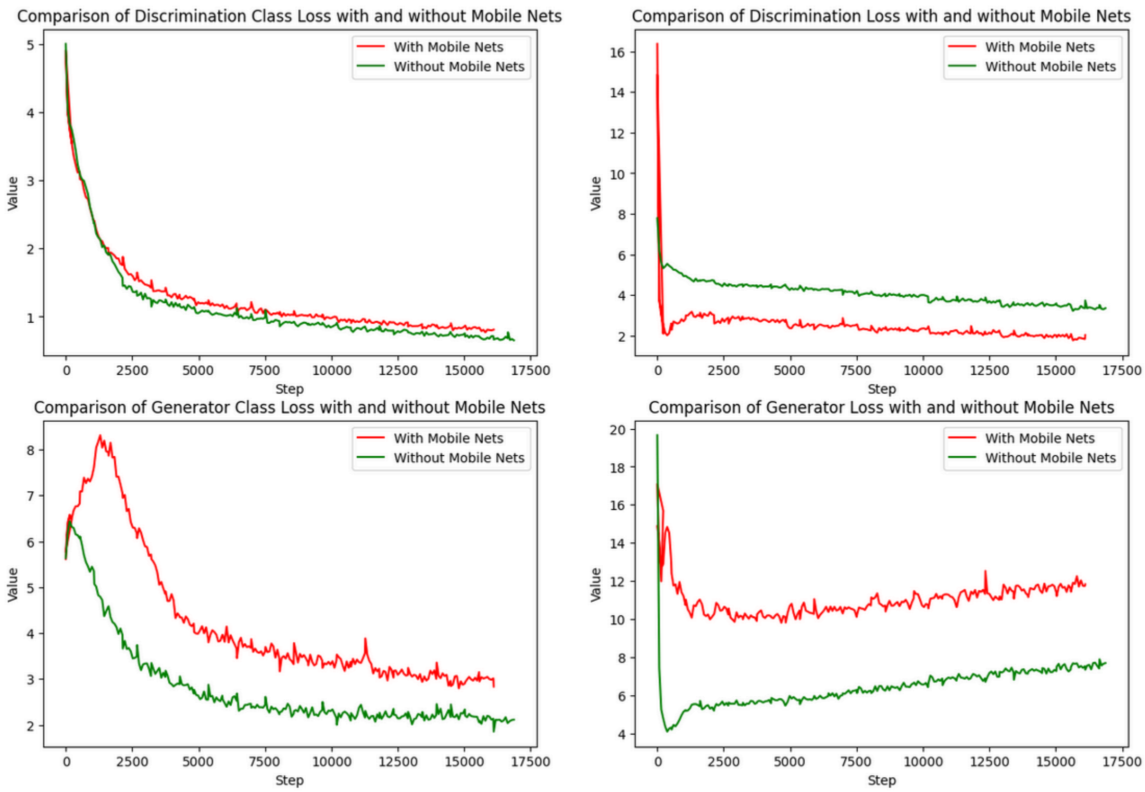


Figure 17 shows the comparison of the result of discriminator and generator loss class with and without MobileNets. MobileNet is an open-source made by Google that was designated for computer vision models and training classifiers. Depthwise convolutions are commonly used in the implementation of MobileNet due to its function as it reduces the number of parameters compared with other networks, which makes the MobileNet architecture produce much faster and can be embedded in smaller model devices. The graph shows two corresponding line plots where the green line represents the result without the extraction of MobileNets architecture. On the other hand, the red line represents the result of the extraction of MobileNets architecture in the model. Moreover, the x-axis represents the value of loss, indicating the number of errors in the algorithm, while the y-axis represents the steps indicating the process validation of the algorithm. However, based on the result, it shows that generation of images without

MobileNets produces better results compared with MobileNets as it is able to generate a larger parameter size. Furthermore, implementing MobileNets in the model might require some parameter adjustment and standard losses need improvement due to the high result of discriminator compared to the generator which makes it bad for generation of aspect mode.

Project Capabilities

The KanonNet model is capable of generating high-quality anime characters from random noise vectors. It incorporates techniques from SRResNet and Spectral Normalization GANs to enhance the quality and stability of the generated characters. It also introduces methods inspired by MobileNets, such as continuous upscaling and downscaling, to extract low-level features and reduce computational requirements.

The model is designed to be computationally efficient, making it suitable for use in resource-constrained environments such as mobile devices. It also includes features for customizing the generated characters, such as selecting different eye and hair types.

Trade-off

Image Quality vs Model Performance

The model performance shows the significant trade offs of generating images for the anime facial structure. Model with MobileNets generates anime facial structure much faster than the model without MobileNets. However, in terms of the quality of the generated images from the anime, the model without MobileNets tends to produce higher-quality images compared to the model with MobileNets. It captures finer details, textures, and nuances of the anime facial structure, resulting in visually appealing and more realistic representations. Moreover, generating anime facial characters without MobileNets often leads to more visually appealing results. The model can produce images that are visually striking, vibrant, and well-balanced in terms of color, composition, and proportions. This enhanced visual appeal can greatly contribute to the overall impact and engagement of the generated anime characters. The idea behind the model without MobileNets produces high quality anime images is because the model architecture of MobileNets are designed to be lightweight and efficient, making them suitable for real-time or resource-constrained applications. However, their architecture sacrifices some level of complexity and capacity compared to larger models. In the context of image generation, this reduced capacity can limit the model's ability to capture intricate details, fine textures, and subtle variations in anime facial features. On the other hand, models without MobileNets, which may have a larger capacity or more sophisticated architectures, can better capture and preserve these details, resulting in higher-quality images. Also, MobileNets are known to be parameter efficient. This means that MobileNets employ techniques like depthwise separable convolutions and network pruning to reduce the number of parameters and computations, thus achieving faster inference times. While this efficiency is advantageous for real-time applications, it can come at the cost of sacrificing some image quality. The reduced parameter count and computation may lead to a loss of information or a simplified representation of anime facial features, resulting in a decrease in image quality. Models without MobileNets, which are not constrained by parameter efficiency, can afford to allocate more parameters and computations to capture finer details and produce higher-quality images.

User Customization vs. Image Quality in AI Anime Generation System

GANs algorithm ensures a high level of artistic consistency across the generated anime images. However, when users introduce customization, the resulting images may vary in terms of style, quality, and overall coherence, depending on the extent and compatibility of the user's preferences. Moreover, user-defined preferences may introduce complexity, making it challenging to maintain the high visual fidelity and coherence achieved by the default mode. While customization allows users to have control over the appearance of the generated anime images, the trade-off with image quality may impact user satisfaction. Users who prioritize customization may be willing to compromise on image quality to achieve their desired personalized anime images, while others may prefer the higher image quality of the default mode. This event occurs due to the nature of the training and generation process. When the generator is trained in the default mode without user input, it relies solely on the learned patterns from the training data. This allows it to generate high-quality images that align with the overall style and aesthetics of the anime genre. The algorithm's parameters are optimized to generate images that are visually appealing and consistent. However, when user customization is introduced, the generator needs to incorporate the user's preferences into the image generation process. This customization adds complexity and constraints to the generation process. The generator must try to satisfy the user's specific requests, which may conflict with the learned patterns from the training data. As a result, the generated images may deviate from the high-quality default mode, as the generator needs to balance user preferences with maintaining visual coherence and consistency.

Chapter 5

Summary, Conclusions, and Recommendations

Summary

In conclusion, the KanonNet model presents a promising solution for automatic generation of anime characters. It combines advancements in deep learning and efficient neural networks to enhance the quality, stability, and variety of generated characters. The model is computationally efficient and customizable, making it a valuable tool for artists, industry specialists, and anime enthusiasts.

Conclusion

The KanonNet model demonstrates the potential of deep learning and efficient neural networks in the creative process. It addresses the challenges in anime character generation and provides a tool that can inspire character design in the animation and video game industry. The model also showcases how advanced technologies can make the creative process more accessible and applicable in resource-constrained environments. Moreover, researchers were able to find the possibility of using GANs in conjunction with Google to generate anime characters in a manner that requires less computation power than the standard GAN model. The model was successfuly able to generate anime characters in a consistent and timely manner which adheres to the general notion of what a character is. Throughout the process, evaluation of the application of the depth wise separable convolutions to the FID score of gans were also conducted. Lastly, project evaluations including the evaluation of GAN with or without MobileNet and system evaluation were also conducted and thoroughly discussed. The study presents the comparison of the performance of the model and the results of system evaluation with respect to ISO/IEC 25010 according to its functional suitability, performance efficiency, and satisfaction with additional metrics such as MSE and InceptionScore. Overall, based on the system result, the proponents of this study conclude that all specified objectives were met.

Recommendations

Future researchers should consider exploring other architectures such as EfficientNet, ResNet, or transformer-based models like Vision Transformers, which could potentially enhance the model's performance. Expanding the dataset to include a wider variety of anime styles and characters could also improve the diversity and quality of the generated images. It would be beneficial to implement a system to gather user feedback on the generated characters, providing valuable insights into the model's performance and areas for improvement. Experimenting with different loss functions could also lead to improvements in the stability of the training process and the quality of the generated images.

While the current model allows for some customization, there's room for more. Researchers could look into ways to allow users to customize more aspects of the characters, such as clothing, accessories, or poses. Considering real-time generation for applications like video games or virtual reality could also be a valuable avenue to explore. As with any AI technology, it's crucial to consider the ethical implications and implement safeguards to prevent misuse. Lastly, staying updated with the latest research in the rapidly evolving field of deep learning will ensure the incorporation of new techniques and best practices into their work.

References

Rezoolab. (2015). Chainerを使ってコンピュータにイラストを描かせる. Qiita. https://qiita.com/rezoolab/items/5cc96b6d31153e0c86bc

Hiroshiba. (2016). Girl Friend Factory - 機械学習で彼女を創る -. Qiita. https://qiita.com/Hiroshiba/items/d5749d8896613e6f0b48

Ledig, C. (2016, September 15). Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. arXiv.org. https://arxiv.org/abs/1609.04802

Aberman, K., Wu, R., Lischinski, D., Chen, B., & Cohen-Or, D. (2019). Learning character-agnostic motion for motion retargeting in 2D. ACM Transactions on Graphics, 38(4), 1–14. https://doi.org/10.1145/3306346.3322999

Miyato, T. (2018, February 16). Spectral Normalization for Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1802.05957

Mescheder, L. (2018, January 13). Which Training Methods for GANs do actually Converge? arXiv.org. https://arxiv.org/abs/1801.04406v4

Goodfellow, I. J. (2014, June 10). Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1406.2661

Radford, A. (2015, November 19). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1511.06434

Xiao, Z. (2021, December 15). Tackling the Generative Learning Trilemma with Denoising Diffusion GANs. arXiv.org. https://arxiv.org/abs/2112.07804

Zhu, J. (2017, March 30). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. arXiv.org. https://arxiv.org/abs/1703.10593

McDonald, K. (2023b, April 15). What is Extreme Programming (XP)? | Agile Alliance. Agile Alliance |. https://www.agilealliance.org/glossary/xp/

Arjovsky, M. (2017, January 26). Wasserstein GAN. arXiv.org. https://arxiv.org/abs/1701.07875

Mao, X. (2016, November 13). Least Squares Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1611.04076

Karras, T. (2017, October 27). Progressive Growing of GANs for Improved Quality, Stability, and Variation. arXiv.org. https://arxiv.org/abs/1710.10196

Odena, A. (2016, October 30). Conditional Image Synthesis With Auxiliary Classifier GANs. arXiv.org. https://arxiv.org/abs/1610.09585

Karras, T. (2018, December 12). A Style-Based Generator Architecture for Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1812.04948

Rumelhart, D. E., Hinton, G. E., & Williams, R. B. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533–536. https://doi.org/10.1038/323533a0

Stanford University CS231n: Deep Learning for Computer Vision. (n.d.). http://cs231n.stanford.edu/

Sandler, M. (2018, January 13). MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv.org. https://arxiv.org/abs/1801.04381

Zhang, H. (2018, May 21). Self-Attention Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1805.08318

Brock, A. (2018, September 28). Large Scale GAN Training for High Fidelity Natural Image Synthesis. arXiv.org. https://arxiv.org/abs/1809.11096

Yazıcı, Y. (2018, June 12). The Unusual Effectiveness of Averaging in GAN Training. arXiv.org. https://arxiv.org/abs/1806.04498

Jin, Y. (2017, August 18). Towards the Automatic Anime Characters Creation with Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1708.05509

Ultraist. (2014). OpenCVによるアニメ顔検出ならlbpcascade_animeface.xml. デー. https://ultraist.hatenablog.com/entry/20110718/1310965532

Datasets & DataLoaders — PyTorch Tutorials 2.0.1+cu117 documentation. (n.d.). https://pytorch.org/tutorials/beginner/basics/data_tutorial.html

Howard, A. G. (2017, April 17). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv.org. https://arxiv.org/abs/1704.04861

StyleGAN-Human Interpolation - a Hugging Face Space by hysts. (n.d.). https://huggingface.co/spaces/hysts/StyleGAN-Human-Interpolation

Bengio, Y. (2009). Learning deep architectures for AI - Université de Montréal. https://www.iro.umontreal.ca/~lisa/pointeurs/TR1312.pdf

Baddeley, A. D. (1997). Human memory : theory and practice (Rev. ed.). Psychology Press.

John, T. (2016). Supporting business model idea generation through machine-generated ideas: A design theory. 2016 International Conference on Information Systems (ICIS)

Jin, J., Dundar, A., & Culurciello, E. (2014). FLATTENED CONVOLUTIONAL NEURAL NETWORKS FOR FEEDFORWARD ACCELERATION. Neural and Evolutionary Computing.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional Neural Networks. Communications of the ACM, 60(6), 84–90. https://doi.org/10.1145/3065386

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2017.195

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision. Computer Vision and Pattern Recognition.

Kulkarni, S. (2023, March 11). How to overcome a creative block. Simplified. https://simplified.com/blog/design/research-proven-ways-to-overcome-a-creative-block/

Maata, P. (2023, May 6). 20+ powerful creativity facts & imagination statistics. DreamMaker. https://dreammakerr.com/creativity-facts-imagination-statistics/

Sankalpa, D., Ramesh, J., & Zualkernan, I. A. (2022). Using Generative Adversarial Networks for Conditional Creation of Anime Posters. https://doi.org/10.1109/iaict55358.2022.9887491

AniGAN: Style-Guided Generative Adversarial Networks for Unsupervised Anime Face Generation. (2022b). IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9541089

Ferjan, M. (2023). 20+ Anime Statistics & Facts: How Many People Watch Anime? (2023). HeadphonesAddict. https://headphonesaddict.com/anime-statistics/#How-many-people-watch-anime