Chapter 6

Multi-objective Cyclic Generative Adversarial Network for Unpaired Image Translation

In the previous chapters, novel evolutionary optimizers have been designed and investigated by addressing the major concerns of ECs to make them suitable for different applications. In continuation, this chapter is entirely focused on addressing the training instability and other convergence pathologies of the cyclic generative model as a large-scale multi-objective optimization problem. A novel training approach has been introduced by exploring the strength of EC for unpaired image translation. Subsequently, in the later part of the chapter, further modifications in terms of a number of fitness functions and gradient-aware intelligent mechanisms along with quantization have been introduced to make our proposal more effective and suitable for IoT-based medical applications.

6.1 Introduction

Image-to-image translation (I2I) aims to transfer images from a source domain to a target domain while preserving the content representations. Because of its wide range of applications in many computer vision and image processing problems, such as style transfer [272], image colorization [273], image super-resolution translation [274], semantic label generation from images, and domain adaptation [275]. I2I has drawn increasing attention and made tremendous progress in recent years. In addition to this, researchers and experts also view the great scope of I2I in the area of medical imaging. A variety of technologies are employed to collect spatially resolved information on organs and tissue in vivo which comprises conventional radiography, magnetic resonance imaging (MRI), and computed tomography (CT). The underlying physical principles are many, resulting in imaging data with varied dimensionality and contrast. This diversity entails a number of diagnostic choices, but it also presents a challenge in terms of the translation of image information across multiple modalities or different acquisitions within one modality. Furthermore, patients find it costly and inconvenient to undergo multiple tests and scans. Moreover, creating a high-quality training set for automatic diagnosis is both costly and time-consuming, especially for rare and critical diseases. Therefore, realistic image synthesis has long been a goal for computer vision. It is noteworthy that before the emergence of Generative Adversarial Networks (GANs)[276], realistic generative modeling was not very successful.

Despite its tremendous success in a wide range of application domains, GANs training remains a complex and demanding task. This is due to the network's adversarial dynamics, which result in a variety of convergence pathologies [12], including mode collapse, vanishing gradient, and exploding gradients, causing training instability [22] and a lack of image diversity. Besides, the requirement of paired images for I2I using GANs is another major concern which is usually not present in the case of old dead paintings and is prohibitively expensive in the medical domain. With their ability to

convert cross-domain images, Cyclic-GANs are a favorable choice to address this issue. A representative image for paired images, as well as unpaired image (cross-domain image) translation, has been shown in Figure 6.1. The major challenges mentioned above are open issues and have become more complex in the case of Cyclic-GAN. Thus, GAN training should be addressed as a large-scale optimization problem. Researchers are constantly striving to solve this challenging optimization problem, presenting various adversarial training algorithms such as reformulating the objective of optimization that controlled the generator parameters and led to attaining the equilibrium point of optimization under defined constraints [277]. Numerous techniques and algorithms can be employed to target a specific domain conversion and achieve better results. Since EC is quite popular in solving complex optimization problems belonging to the NP-hard class and multi-objective optimization due to its inherent characteristics, the researcher also experimented EC with GAN, resulting in Evolutionary GAN (E-GAN) [180] with enhanced generating ability. Similarly, ideas such as coevolution, neuroevolution, and evolutionary pressure have also experimented with GAN. These concepts, however, are being experimented with using GAN. While none of these algorithms are comprehensive enough to work with a diverse range of data sets and applications. Further, research on the utilization of EC with Cyclic-GAN is still in its embryonic stage.

Therefore, with inspiration from E-GAN, in this chapter, we first propose a novel Evolutionary Multi-objective Cyclic-GAN (EMOCGAN) to address the above-stated challenges related to Cyclic-GAN training for the I2I translation. In this work, we have also introduced a new approach for model training by integrating the concept of EC, multi-objective optimization, and Cyclic-GAN along with different selection mechanisms. To overcome local optima stagnation, metropolis acceptance criteria and Pareto-based selection on two scores (objective functions) are utilized initially. Evolutionary concepts in training helped to address the instability and mode collapse problems. Extensive experiments on real-world image datasets show that the EMOCGAN outper-

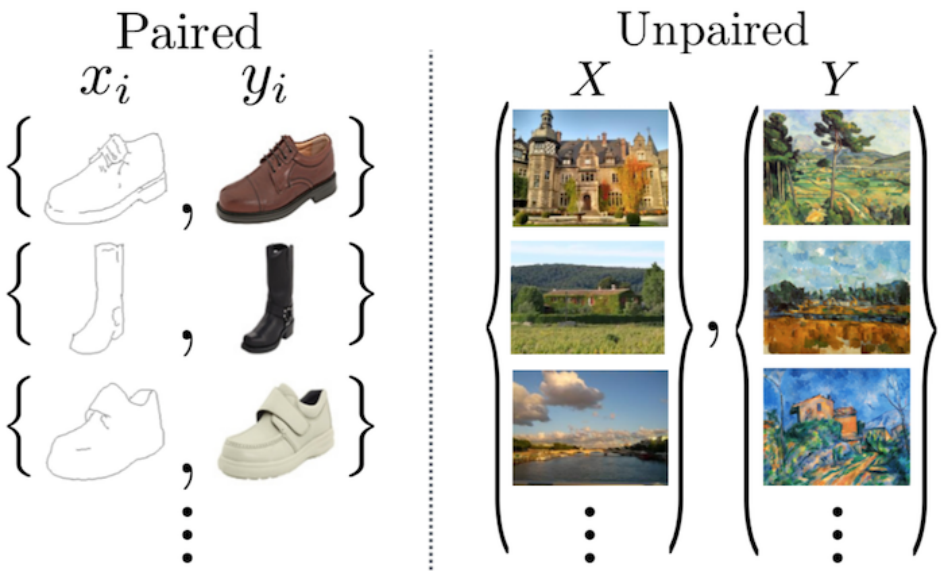


Figure 6.1: Unpaired v/s Paired image-to-image translation

forms the state-of-the-art method in terms of visually realistic appearance and retaining background information as well as salient objects. Quantitative comparisons of our EMOCGAN with the Cyclic-GAN also show significantly better scores obtained by our model, as indicated by Structural Similarity Index (SSIM) and Universal Quality Index (UQI). The model demonstrated the best efficacy in terms of SSIM on Apple↔Orange, while it shows higher UQI values for Monet↔Picture and Summer↔Winter datasets.

After an in-depth analysis of EMOCGAN, it has been observed that the generating ability can be enhanced further, which leads to the incorporation of one more objective function in addition to the previous one for Pareto based selection scheme, and for simplicity, we named the model EMCGAN. Further, the industrial revolution 4.0 and advancements in technology led to the development of smart services. Now, cloud computing, IoT, big data, AI, wearable devices, and robots frequently become an essential part of various applications such as healthcare transformed into healthtech. Also, it raises the demand for automated IoT-based applications. Therefore in a similar spirit, the novel Quantized Evolutionary gradient aware Multi-objective Cyclic-GAN (QEMCGAN) model with few other significant modifications has been proposed in the next

part of the chapter. Model quantization was used owing to its suitability for low-cost IoT-based applications. In addition to this, a novel intelligent mutation selector that considers the gradient of training has been proposed, as quantization has a significant impact on model performance. This intelligent mechanism also changes the previously random decision between the crossover and training algorithm. While maintaining the strength of the previous model, we have been able to achieve more visually realistic images, even in the case of cross-domain translation for medical images, along with other unpaired natural image translations. It was found that even when the model size was halved, QEMCGAN showed decent performance with the baseline approach, making it more efficient.

6.2 Motivation and Contribution

It is evident from the prior discussion that the limitations involved in Cyclic-GAN training are mode collapse and instability issues which have to be dealt with as a complex large-scale optimization problem. In addition, the requirement for a large data set comprising paired images is another bottleneck of deep models. Moreover, in the case of rare diseases, collecting a large amount of paired data is a difficult task necessitating domain expertise. In such cases, cloud computing and IoT-based applications can be used to collect images through various medical institutions. However, data privacy remains a major concern. To address privacy issues in such a framework, a generative model can generate visually appealing images that can be used for knowledge distillation without revealing the original data. However, multi-modal medical imaging is another potential method through which clinical decision-making can be enhanced. Since collecting data from the same patient using different imaging techniques, such as CT scans and MRIs, is often impractical due to the additional time required for multiple scanning sessions at a certain cost or limited access to imaging techniques, cross-domain medical image translation shows a promising future (different imaging modalities referred to as the

domain). The main objective of this work was to develop a model that can overcome the training issues commonly encountered in Cyclic-GANs, such as mode collapse and prolonged training times. The wider scope and challenges motivated us to address the aforementioned issues in the context of multi-objective optimization problems by leveraging the strength of ECs. To the best of our knowledge, this is the first effort made for Cyclic-GAN with EC.

The major contributions of this study are as follows:

1. Three different novel frameworks, namely EMOCGAN (2-objective) with random selector, EMCGAN (3-objective) with intelligent selector, and finally, quantized version- QEMCGAN, have been proposed.

2. The training process of frameworks has been formulated as a multi-objective optimization problem with two different scores in EMOCGAN and three different scores in EMCGAN and QEMCGAN as fitness evaluation metrics. This optimization problem is solved using NSGA-II. Evolutionary training alongside two different selection schemes was used to address the mode collapse limitation.

3. With randomization, two different mutation processes were incorporated (first population crossover and second fine-tuning parent). In addition, Pareto-based ranking and simulated annealing-based concepts were used as selection schemes to maintain convergence and diversity.

4. Instead of discarding the entire model, the two worst scoring models (at most) were selected, which helped in addressing the issue of local optima stagnation in accordance with the Metropolis acceptance criteria.

5. Several loss functions were considered to improve the generation quality, among which heuristic loss was modified to render it suitable for Cyclic-GANs, which facilitated solving the vanishing gradient problem.

6. We proposed a novel intelligent mutation selector for EMCGAN and QEMCGAN that considered the gradient of training, as quantization had a significant impact on model performance. This intelligent mutation selection changed the previously random decision between the crossover and training mutation.

7. We successfully demonstrated the effectiveness of the proposed methods by applying them to three real-world natural images as well as medical image data sets for cross-domain conversion.

6.3 Theoretical Background

This section gives fundamental definitions that are closely related to the proposed study. Boldface notation is used to represent vector-valued variables.

Definition 6.1 *Multi-objective optimization (MOO): MOO is an optimization problem that includes many conflicting objective functions which need to be minimized, maximized, or both simultaneously, such as aircraft design, model structure learning and so on [278]. It can be mathematically written as follows:*

$$\begin{aligned} Min\ \Psi(\boldsymbol{a}) = (\psi_1(\boldsymbol{a}), \psi_2(\boldsymbol{a}), \cdots,\ \psi_k(\boldsymbol{a})) \\ s.t.\ \boldsymbol{a} \in \Omega \end{aligned} \tag{6.1}$$

where $\boldsymbol{a}= (a_1, a_2, ..., a_d) \in \Omega$ *is a decision vector in a search space (decision variable search space) denoted by* Ω. $\Psi : \Omega \rightarrow \mathbb{R}^q$ *denotes the function mapping of decision vector* $(d - dimenaion)$ *to the objective space with* $(q - dimension)$.

Definition 6.2 *Pareto dominance : To compare the merits of two distinct solutions* $\boldsymbol{a}_1$, $\boldsymbol{a}_2$ *of a MOO problem, the Pareto dominance relation is computed.* $\boldsymbol{a}_1$ *dominates another solution* $\boldsymbol{a}_2$ *i.e* $(\boldsymbol{a}_1 \prec \boldsymbol{a}_2)$ *iff*

$$\begin{cases} \forall i \in \{1, 2, \cdots, k\}, & \psi_i(\boldsymbol{a}_1) \leq \psi_i(\boldsymbol{a}_2) \\ \exists j \in \{1, 2, \cdots, k\}, & \psi_j(\boldsymbol{a}_1) < \psi_j(\boldsymbol{a}_2) \end{cases} \tag{6.2}$$

Two solutions $\boldsymbol{a}_1$ *and* $\boldsymbol{a}_2$ *in* Ω *are said to be* ***non-dominated (ND)*** *with respect to each other if neither* $(\boldsymbol{a}_1 \prec \boldsymbol{a}_2)$ *nor* $(\boldsymbol{a}_2 \prec \boldsymbol{a}_1)$ *.*

Definition 6.3 *Pareto optimal solution (POS): A feasible solution* $\boldsymbol{a}^\star \in \Omega$ *is POS iff* $\boldsymbol{a}^\star$ *is a* ***ND*** *solution i.e* $\nexists\, \boldsymbol{b} \in \Omega$ *such that* $\Psi(\boldsymbol{b}) \prec \Psi(\boldsymbol{a}^\star)$*. A set of all possible POS in* Ω *called a Pareto-optimal set (PS), and Pareto Front (PF) is represented as* $\{\Psi(a)\,|a \in (PS)\}$*.*

6.4 Proposed Methodology

In this section, we will first present the Cyclic-GAN and then introduce our proposed EMOCGAN, followed by the EMCGAN, which employs an intelligent mutation selector, and finally, its quantized version, QEMCGAN.

6.4.1 Cyclic Generative Adversarial Network

The GAN network implemented is similar to the network provided by [279] but differs in the architectures of the generator and discriminator.

6.4.1.1 Generator

The generator input is a series of randomly generated numbers called latent samples. We have two generator networks based on ResNet architecture containing ResNet links. The first generator converts the images of the first domain to the second domain, and the other converts the images of the second to the first. In the generator, we compared the performance of various networks, and the architectures with the best performance

in image-based task was selected. This inference is supported by the results shown in Section 6.5.

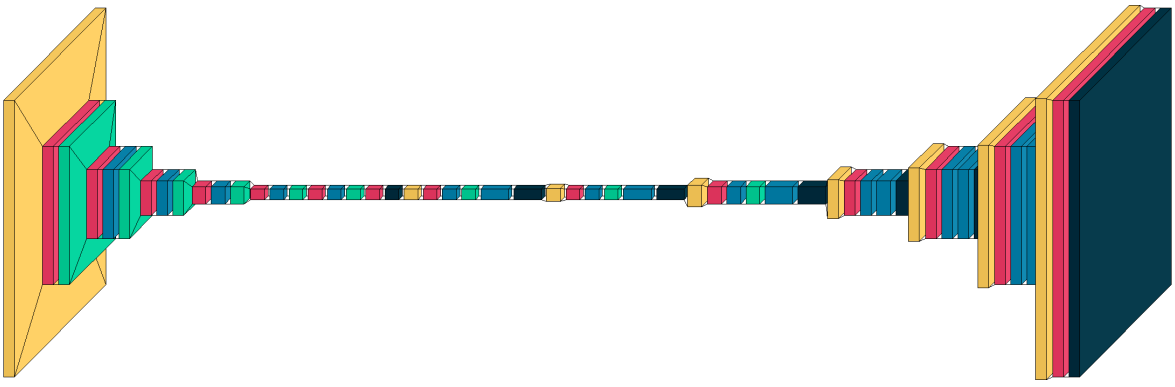
6.4.1.2 Discriminator

Supervised learning is used to train the discriminator network that works as a classifier to predict whether an image is real or not. We have used a simple CNN-based architecture. In the case of Apple ↔ Orange data, the first discriminator classifies apples, and the other classifies oranges into fake and real. Similarly, the second discriminator does the reverse.

6.4.1.3 Architecture details

We have followed a similar network architecture for all three proposals. In this, the discriminator performs the classification operation by predicting the input image as real or fake. It has a simple architecture, and the details of the discriminator networks are presented in Figure 6.2 (a). At the same time, the generator's architecture is intricate due to its functionality. These models are made up of small network blocks which are linked in a parallel fashion to form resnet links between each other. A block is the most

(a) Discriminator network



(b) Generator network used in the proposed models

Figure 6.2: Representation of GAN

basic form of a CNN, consisting of the following layers: convolutional 2D layer, batch normalization layer, and Leaky ReLU in the activation layer, where batch normalization deals with the exploding gradient problem. The obtained results are combined due to the parallel connection. Thereafter to obtain images of the original shape, ReLU activation is applied to the combined results, and the blocks are recurring in reverse order, with the Conv2D layer superseded by Conv2D_Transpose. A layered view of the generator is shown in Figure 6.2 (b). This configuration enables to encapsulate new losses, as mentioned in virtual restaining work [279]. In this framework, various losses are considered, which we will discuss in a later section.

6.4.2 EMOCGAN

The EMOCGAN implemented here is a variant of the original E-GAN [180]. We have modified it for the cyclic variant by integrating the powerful concepts of MOO and ECs along with other suitable modifications for making the training process more effective. The population used is the whole Cyclic-GAN model, which consists of two generators and two discriminators, i.e., one sample in the evolutionary training population consists of two generators and two discriminators in the Cyclic-GAN configuration. These samples are trained, fine-tuned using evolutionary concepts, and filtered in every iteration. The best generator pair yielding the highest fitness score is selected as the optimal generator pair; here, the selection is formulated as MOO. The evolutionary training consists of three steps, i.e., mutation, evaluation, and selection. Each step is described in further sections. The overview of the model is shown in Figure 6.3.

6.4.3 EMCGAN

The EMCGAN is also an integrated framework that uses the power of EC and MOO in solving large-scale optimization problems for Cyclic-GAN same as EMOCGAN. This proposal has a similar architecture and other schemes used in EMOCGAN except for a number of fitness functions and an intelligent selection scheme instead of random.

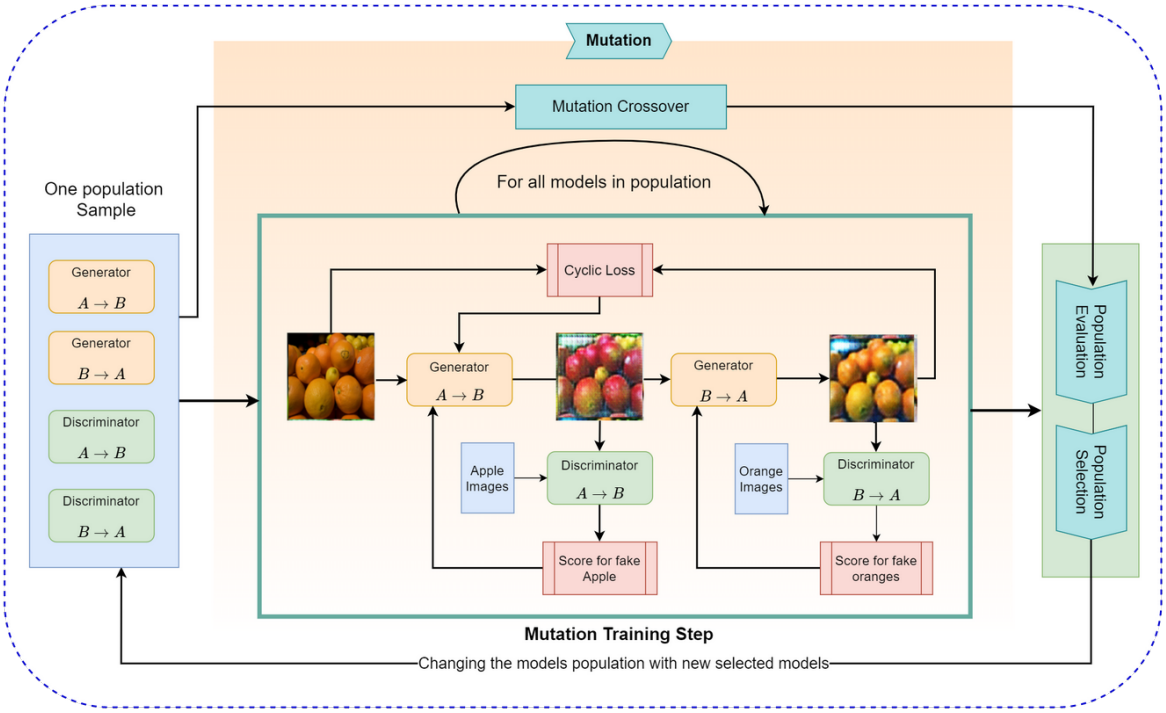


Figure 6.3: Overall framework

The working of the proposal can be described in the context of medical cross-domain translation. Here, the first generator is used to translate the image of the first context to the second (for example, MRI $\rightarrow$ CT), while the latter translates the second context to the first (CT $\rightarrow$ MRI). The discriminator in the framework is trained using supervised learning, which is based on a simple CNN. It acts as a classifier, taking an image as input and providing a probabilistic estimate of whether or not the image is real. The two generators are coupled together to form a unit that serves as the first and second generators in the Cyclic-GAN and their respective discriminators.

Each sample in the population used for evolutionary training consists of the whole Cyclic-GAN model, and an evolutionary algorithm is then used to train and fine-tune this population. Additionally, samples are filtered after each iteration. The optimal generator pair is determined by selecting the best generator pair with the highest fitness score. Here, we have formulated a selection mechanism as a MOO problem, and mutation, evaluation, and selection are the three steps in evolutionary training. In this

variant, we further improved the EMOCGAN, and instead of two objective functions, we have incorporated three objective functions in MOO to improve the diversity and quality of generating the image. In the original version, the decision between crossover and mutation schemes was made at random. In contrast, in EMCGAN, we proposed an intelligent mutation selector and also applied quantization, resulting in our proposed quantized version of EMCGAN, which we will discuss in Section 6.4.4.

6.4.3.1 Mutation-operation in Training

The most simple and obvious step would be to use mutation through training. Still, we have also used mutation through the crossover to avoid sample models from moving toward the common minimum, which would result in all models being equivalent, and the aim of preserving the population is lost. The mutation crossover produces offspring models, which we refer to as the child population, by shifting all of the weight by some factor.

To obtain samples for the next iteration, we used two strategies for mutating the parent samples. These strategies are chosen at random, and therefore the selected approaches are unpredictable for each sample in each iteration. This randomness is used to force the model to attain global optima instead of sub-optima, thus lowering the model's hyperparameter sensitivity.

1. **Implementation of the mutation-crossover operator to the parent framework:** The parent framework's weights are randomly modified in this step to build the next generation of the model. Because it is not optimizing anything, this method is fairly random and may not generate the best results, but it gives a sporadic shift from the main training to make it less prone to local minima problems. Thus, the recurrence of this procedure decreases as the number of epochs increases. Another important aspect of performing this operation is that if we do not have this shifting, we may end up with several models being quite similar.

Algorithm 6.1: Mutation By Training

Input: $Data_{muta}$, population (model population)

Output: childPopulation (child model population for next iteration)

```
childPopulation ← emptyList()
for model ← model_1 to model_N from population do
    M_child ← model.copy() ▷ M_child is Child Model
    while epochs ≤ max_epochs do
        L_cyclic, L_heuristic, L_identity, L_DS ← loss()
        L_Generator, L_Discriminator ← mainLoss()
        M_child ← train(M_child, Data_muta, losses...)
    end
    childPopulation.add(M_child)
end
return childPopulation
```

2. **Model training on data to obtain better model than parent version:** In order to enhance global consistency, the parent framework is modified by acclimating the model parameter with different losses. Using Adam as the optimizer, this framework is trained for around two to three iterations. Algorithm 6.1 describes the steps for the entire training process. We have used cyclic loss($\pounds_{cyc}$), identity loss($\pounds_{id}$) as described in [279] along with some additional losses namely diversity sensitivity loss($\pounds_{DS}$) and heuristic loss ($\pounds_{heu}$). Each type of loss has contributed in overall loss function ($\pounds(G_a, G_b, D_a, D_b)$), where G_a and G_b represent the generators for conversion of $(a \rightarrow b)$ and $(b \rightarrow a)$ respectively while D_a and D_b are corresponding discriminators. The following equations present the considered loss functions.

- **Cyclic Loss** ($\mathscr{L}_{cyc}$)**:** Cyclic loss is used for GAN in performing unpaired I2I translation. Cyclic-GAN uses encoder-decoder operation as a cycle ($X \rightarrow$

$Y \rightarrow X'$) for generating $X \approx X'$ in each subsequent cycle.

$$\mathscr{L}_{cyc}(G_a, G_b) = \mathbb{E}_{x \sim Pdata(x)}||x_{rec} - x||_1 + \mathbb{E}_{y \sim P_z(z)}||y_{rec} - y||_1 \tag{6.3}$$

where x_{rec} and y_{rec} are the recreation of the images using the generator in the model. For instance the x_{rec} is obtained by $G_b(G_a(x))$ and similarly y_{rec} is obtained by $G_a(G_b(y))$. Throughout the work, we have used these notations for addressing generators and discriminators as well. Similarly, x denotes input, and z corresponds to noise data passed, respectively.

- **Identity Loss ($\mathscr{L}_{id}$):** The $\mathscr{L}_{id}$ was incorporated to retain the color composition, especially when the input image (domain 1) is extremely near to the output image (domain 2) [279].

$$\mathscr{L}_{id}(G_a, G_b) = \mathbb{E}_{x \sim p(x|c)}[||y_{fake} - x||_1] + \mathbb{E}_{y \sim p(y|c)}[||x_{fake} - y||_1] \tag{6.4}$$

x_{fake} and y_{fake} are the fake images generated using the generator in the model. For instance the x_{fake} is obtained by $G_a(x)$.

- **Diversity Sensitivity Loss ($\mathscr{L}_{DS}$) :** It essentially eliminates the mode collapse problem that plagues the vanilla GAN. According to this study [280], GAN generates a variety of images using this model. This loss function's value is proportional to the diversity of the generated images; thus, we want it to increase.

$$\mathscr{L}_{DS} = \mathbb{E}_{z_1, z_2}\left[min(\frac{\|G(x, z_1) - G(x, z_2)\|}{\|z_1 - z_2\|}, r)\right] \tag{6.5}$$

- **Heuristic Loss ($\mathscr{L}^{heu}_{G_a, G_b, D_a, D_b}$):** The mutation loss function, which we incor-

porated in our framework, combined all the loss functions mentioned above and the heuristic loss function originally proposed in E-GAN. Before combining, we modified the heuristic loss function for this study to make it suitable for Cyclic-GAN developed in the first stage. Because of this addition, the vanishing gradient issue is avoided.

The final heuristic loss function used is:

$$\begin{aligned}\mathscr{L}^{heu}_{G_a,G_b,D_a,D_b} = -0.5 \times \mathbb{E}_{z\sim p_z}[log(D_a(y_{fake})) \\ + log(D_b(x_{fake}))] + \mathscr{L}_{cyc}\end{aligned} \tag{6.6}$$

These losses are given different weights based on their contribution to the overall model when the model is evaluated with only that loss. The combined loss function below shows their percentage participation in generator loss.

$$\Upsilon^* = arg \min_{G_a,G_b} arg \max_{D_a,D_b}(\mathscr{L}(G_a, G_b, D_a, D_b)) \tag{6.7}$$

$$\begin{aligned}\mathscr{L}(\mathtt{G}_a, \mathtt{G}_b, \mathtt{D}_a, \mathtt{D}_b) = (\lambda_1) \times \mathscr{L}_{GAN} \\ + (\lambda_2) \times \mathscr{L}_{cyc} + (\lambda_3) \times \mathscr{L}_{id} \\ + (\lambda_4) \times \mathscr{L}_{heu} - (\lambda_5) \times \mathscr{L}_{DS}\end{aligned} \tag{6.8}$$

where, $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ are scalar constant with 0.53, 0.37, 0.09, 0.005 and 0.005 respectively.

6.4.3.2 Fitness Evaluation

The population is evaluated using three evaluation metrics treated as fitness functions during the model training process and is a good evaluation metric for GAN in general. They are as follows:

1. **FID (Fréchet Inception Distance):** FID score is proposed by [281] that computes the distance between feature vectors in real and generated images. Lower scores indicate that the two images have comparable statistics (more similar). It is shown as follows:

$$FID = \|\mu_{real} - \mu_{gen}\|^2 + Tr(\sum_{real} + \sum_{gen} -2(\sum_{real}\sum_{gen})^{\frac{1}{2}}) \tag{6.9}$$

where (μ_{real}, $\sum_{real}$) is (mean, standard deviation (SD)) of real distribution and (μ_{gen}, $\sum_{gen}$) are mean and SD of generated distribution.

2. **IS (Inception Score):** IS [282] is a well-known metric that determines how realistic the output of a GAN looks. The score considers two factors at the same time - image quality and image diversity. The score will be high if both of these are true. It is shown below.

$$IS(G) = [exp(\mathbb{E}_{x\sim p_g})D_{KL}(p(y|x)||p(y))] \tag{6.10}$$

here ($x \sim p_g$) denotes that an image (x) is sampled from (p_g,$p(y|x)$), which represents the conditional class distribution and marginal class distribution is written as ($p(y) = \int_x p(y|x)p_g(x)$). The KL-divergence is represented as $D_{KL}(R||S)$ between two distributions R & S.

3. **TVD (Total Variation Distance):** TVD is a metric used to evaluate the model's diversity and cover the real image distribution, thus preventing the model from the mode collapse issue.

$$TVD = \frac{1}{2}\sum |freq_{real} - freq_{fake}| \tag{6.11}$$

Using these evaluation metrics, the Pareto ranking method [283] ranks the various

Algorithm 6.2: Evaluation

Input: $Data_{fitting}$, population (Population including childModels)

Output: populationSCore

```
populationSCore ← emptyList()
for model ← model_1 to model_N from population do
    fidScore ← fid(model)
    inceptionScore ← Inception(model)
    tvdScore ← TVD(model)
    score ← {-fidScore, inceptionScore, tvdScore}
    populationScore.add(score for model_i)
end
return populationScore
```

models that belong to a population. In the case of EMOCGAN, two scores (FID, IS) have been utilized, while other proposals used all three. It is worth noting that for simplicity and better understanding, Algorithms have been described as per EMCGAN. The steps are described in Algorithm 6.2.

6.4.3.3 Selection

In the Evaluation phase, a Pareto ranking-based selection and random selection are performed. Algorithm 6.3 provides a detailed description of the Selection phase. The algorithm has three main phases, where Phase-I and Phase-II come under Pareto ranking-based selection (NSGA-II), and Phase-III belong to random selection.

1. **Pareto ranking-based selection:** This step is explained in two phases, as described in Algorithm 6.3. In phase-I, We calculate the model population's scores. Three scores(IS, FID, and TVD) were computed for each model on the population. We further employed NSGA-II on these three scores. It ranks each individual based on these scores without considering score priority. Essentially, we obtained Pareto fronts by taking crowding distance into account for each score

Algorithm 6.3: Selection

Input: $Score_P$ (scores for models in population), models (model population including childModels)

Output: $resultModels$ (filtered model by Selection)

▷ Calculating the scores - Phase-I

resultModels ← emptyList()
fronts ← NSGAII($Score_P$) ▷ NSGAII on the scores for rankings
cDists ← calcCrowdingDist($Score_P$) ▷ Calculates crowding distance

▷ Selecting best Models - Phase-II

selected ← 0
i ← 0
while *True* **do**
    resultModel.add(fronts[i]) ▷ Considering front[0] is best front
    selected ← selected+fronts[i].length ▷ Adding all models in fronts[i] to result
    if *selected + length(front[i+1]) ≥ max_selection* **then**
        break ▷ End Loop
    end
end
arrangeFront(fronts[i], cDist) ▷ Arrange front based on cDist
while $selected \leq max_{selection}$ **do**
    resultedModels.add(fronts[i].best)
    selected ← selected+1
end

▷ Random Selection - Phase-III

selected ← 0
$max_{selection} \leftarrow max_{random}/iterations$
while $selected \leq max_{selection}$ **do**
    model ← randomModel(fronts) ▷ Selectes random model from fronts
    resultModel.add(model)
end
return resultModel

in the list. In this phase, the models having the highest rank are chosen as the best models from the population space.

Further, in phase-II, we keep selecting models by including fronts based on the ranking mentioned earlier until the total number of models selected does not surpass the maximum threshold for selection. Then, based on the crowding distance, we arrange the value in the front that we are working on and select the remaining models. In each iteration, five models that have been determined as the best acceptable threshold are chosen in particular since this threshold stops the worst model from progressing to the next iteration while preserving a sufficient number of models.

2. **Random selection:** This step is shown in Phase-III of Algorithm 6.3. Here, we introduced a random model selection in this case to build the selection process less stringent by giving less optimum models a shot as well. We eliminate other models that may consistently beat the selected models in subsequent rounds by just picking models with the top score rank. Simulated annealing is a similar concept. Metropolis acceptance criteria also support this type of selection. We changed the maximum selection value to be inversely proportional to the generation or iteration in this phase to reduce the random selection value as the iteration increased. It ensures that when training is complete, we have all of the best models rather than just a few that were chosen at random.

6.4.4 Quantized EMCGAN

In the context of current demand, the ultimate objective is to make our EMCGAN model capable of working in low-performance or low-space scenarios such as IoT. In this section, we will introduce the quantization into the EMCGAN model that we discussed previously and explore its impact on the model. We tried two different methods to reduce the size of our final generator model.

1. **Teacher-Student model:** This is inspired by model compression via distillation and quantization work [284]. The framework's basic idea is that we take the original model that we want to compress as the Teacher model and then train another neural network (Student model) to fit the Teacher model's results. As a result, the lower-end model will be roughly able to produce similar results while reducing the size dramatically. We made a lower-end version of the model architecture of our EMCGAN model and trained it using the Teacher-Student framework. Because it is a different framework, one of the key advantages of this method is that it can compress any Cyclic-GAN model.

2. **Model quantization:** We modified the existing EMCGAN network in this method so that it can be quantized after training. We used the quantization-aware training framework present in the TensorFlow model optimization package [285] and quantized the convolution and batch-normalization layer. We chose these layers because they contain most of the model's parameters, and by quantizing them, we can drastically reduce the model's size. The trained models were then extracted and converted to a tflite model, which reduces their size and significantly improves their performance on IoT devices. One of the main purposes of the batch normalization layer was to ensure that the gradient did not explode, which means that their value did not grow to huge numbers; to prevent this, we added a gradient check to the model training.

Both methods described above were trained and tested, but none of the models produced satisfactory results. In the first model, due to the high complexity of our EMCGAN model, lower-end student models were not able to learn the EMCGAN (Teacher) model, and in the second, because the convolution layer is the core layer of our model, quantizing that has a significant impact on the model's performance. As a result, the model is not being trained. The final model, which we are using to quantize the base EMCGAN model, employs both training-aware quantization and post-training quanti-

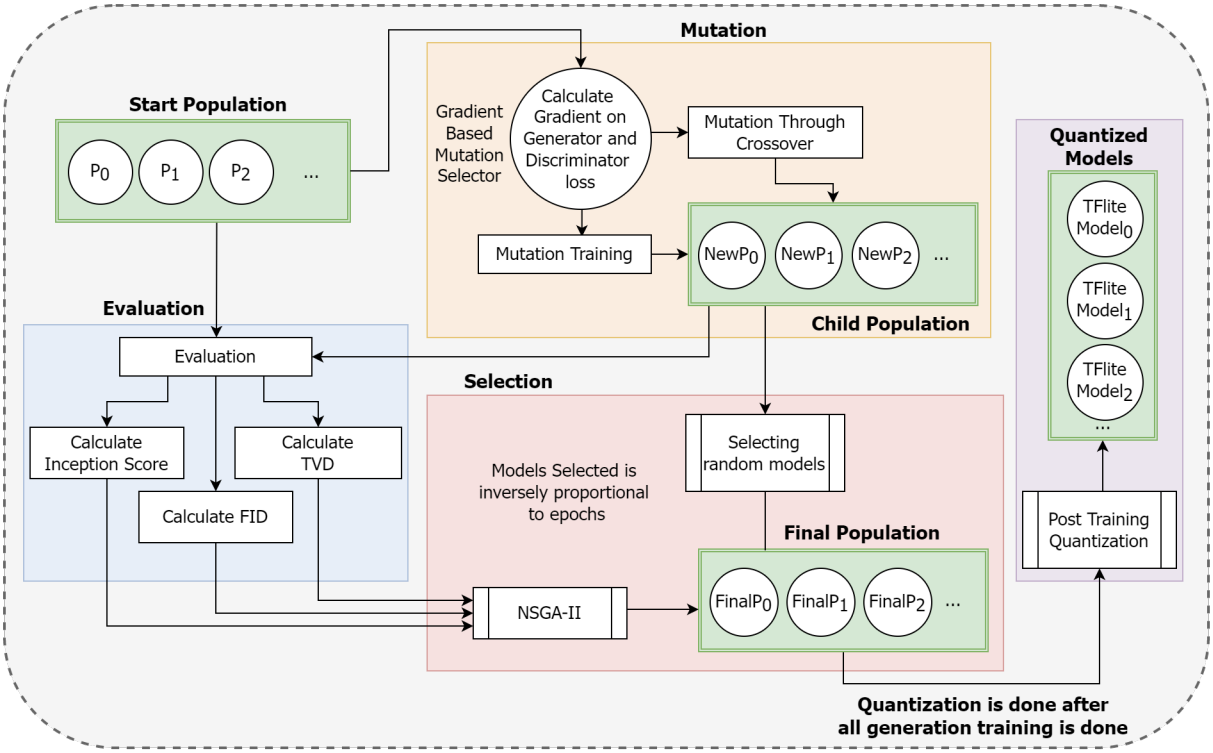


Figure 6.4: Workflow of the proposed QEMCGAN

zation to produce a competitive model with a smaller size. Both of these are described below. The overall framework remains the same, except for the quantization steps we took, but these steps do not change the way the model works. The workflow of the proposed QEMCGAN is presented in Figure 6.4.

6.4.4.1 Training Aware Quantization

As quantizing, the convolution layer in the model is not producing good models. We removed quantization from the convolution layer and only used it in batch normalization. The increase in model size caused by this step is compensated by the post-training quantization that we added (discussed in the next section). This way, we are not losing too much in accuracy while still keeping the model size small enough. This quantization effort can affect the final model accuracy score very drastically; therefore, to address this, we added some intelligent mutation selection, which changes the random decision we were making between the crossover and training mutations to a much more intelli-

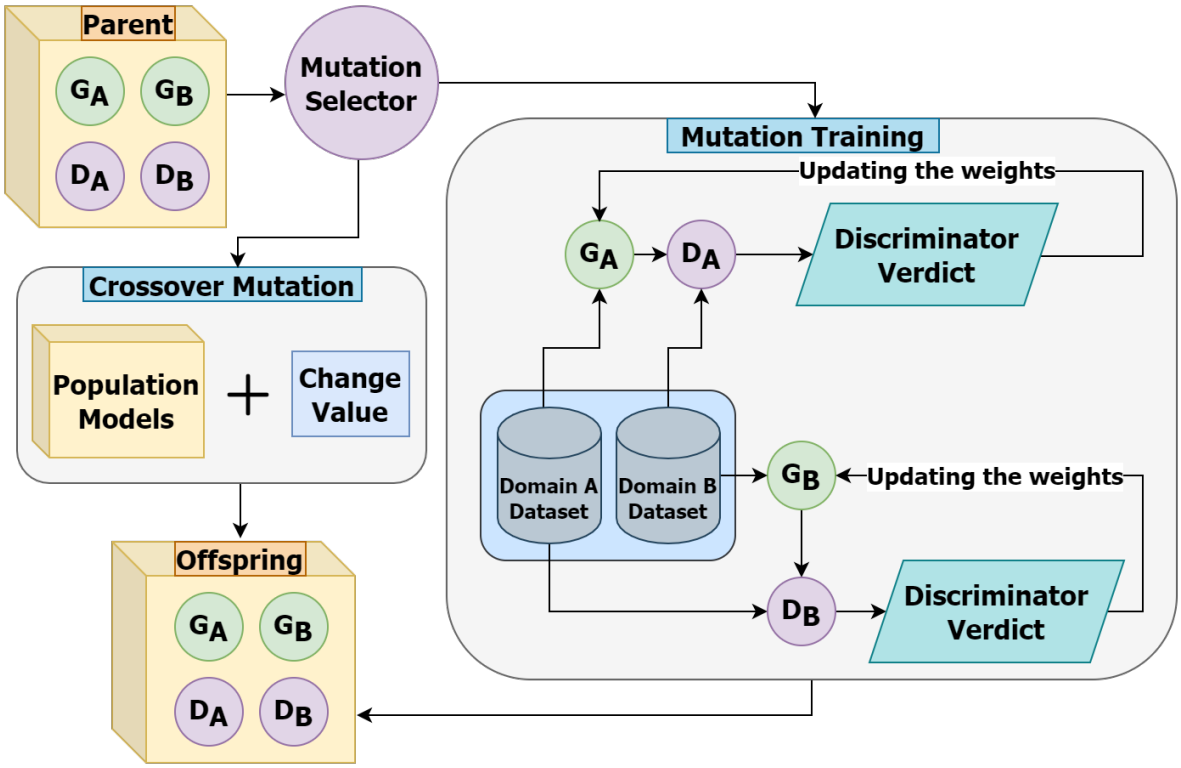


Figure 6.5: Mutation step of the evolutionary algorithm

gent mutation selection that takes the gradient of training into account.

Intelligent mutation selector: This selector is added just before the mutation step to determine which mutation operator to use. In this step, we calculate the gradient of the model on the mutation data and decide which mutation step to take based on the mean gradient of the generator and discriminator. This mutation selector is designed to compensate for the accuracy loss caused by the quantization. Figure 6.5 depicts the mutation step for the entire evolutionary algorithm, while Algorithm 6.4 explains the mutation selector steps.

6.4.4.2 Post Training Quantization

The proposed method also employs post-training quantization as a final step to reduce the trained model to a smaller model with the same effectiveness. This is done because

Algorithm 6.4: Mutation Selector

Input: data, model

Output: *choice* (choice of mutation step)

```
tempModel ← clone(model)
grad_G ← 0
grad_D ← 0
selected ← 0
▷ Selecting best Models
i ← 0
while True do
    batch ← nextBatch(data)
    loss, grads ← trainModel(tempModel, batch)
    grad_D ← grad_D + grads['discriminator']
    grad_G ← grad_G + grads['generator']
end
grad_G ← grad_G / numBatches
grad_D ← grad_D / numBatches
if grad_D ≥ trainLimit and grad_G ≥ trainLimit then
    return 'crossover'
    else
        return 'train'
    end
end
```

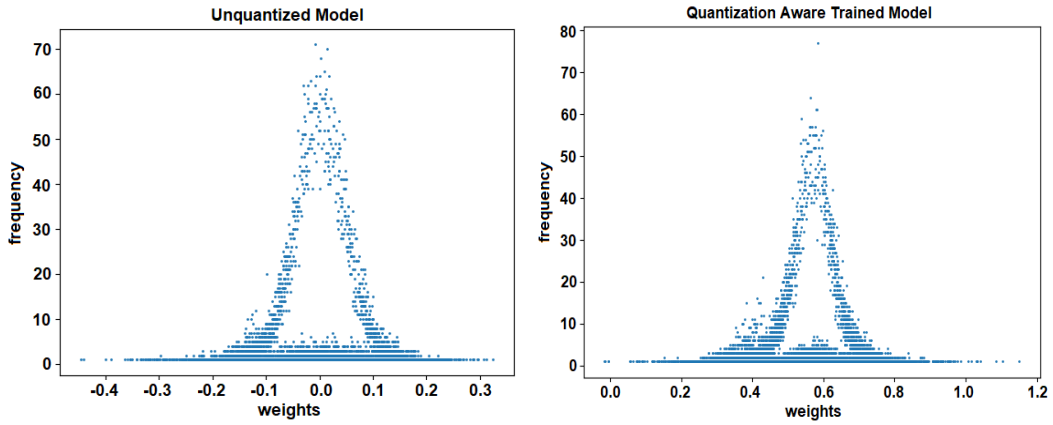
the size of the model is only reduced after it has been trained when it is converted to the tflite version. To reduce the model size, we changed the datatype of the weight to float16 and used the tflite built-in ops, which are faster than standard TensorFlow operations.

Figure 6.4 depicts the entire process, from evolutionary training to post-training quantization. Because this is a model workflow, model changes made for quantization-aware training are not shown.

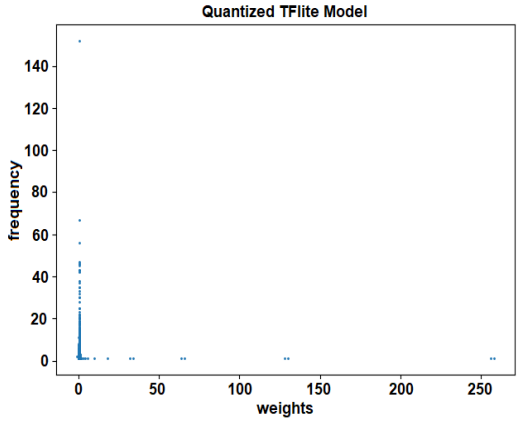
6.4.5 Analysis of Quantization

The method described above is capable of reducing the model size by more than 50% while maintaining model performance almost similar to the original model. The original model size was 54 MB, which has now been reduced to 27 MB. Additionally, because the model is now a tflite model, it can be easily ported to various Android-based devices. Despite the reduced model size, the model performs pretty well on some general domain transfer datasets, even outperforming the original model in some cases.

The plots in Figure 6.6 show the weighted value distribution of the model's generator. These plots show that the unquantized model's weights are very spread and negative, necessitating an extra bit to store that information. In contrast, the quantized



(a) Unquantized EMCGAN Model (b) Quantization Aware Trained Model



(c) Quantized Tflite Model

Figure 6.6: Generator's weight value distribution

weights are less spread, necessitating a smaller data type. All of these are positive. We then used post-training quantization to narrow the range to 0-256 (int8), which clearly shows the effect of quantization on the model's weights.

6.5 Experimental Results

All experiments are carried out on a GPU node built with a 2.4GHz Intel-Xeon Skylake 6148 CPU, 192GB RAM, and a 16GB NVIDIA V100PCIe accelerator card. The proposed model was implemented using Keras with TensorFlow as the backend.

6.5.1 Dataset Description

We experimentally validated our proposed frameworks on three real-world image benchmark datasets (Apple and Orange), (Winter and Summer), and (Picture and Monet) provided in [272] for unpaired I2I translation. The tabular description of the datasets is also detailed in Table 6.1. In addition to this, we have also validated our proposals for medical image domain conversion by analyzing three different challenging datasets- MRI↔ CT, MRI-T1↔ MRI-T2, and stain translation of pathology images. Section 6.5.7 elaborates on the details of the medical dataset utilized in this work.

Table 6.1: Datasets description

Dataset	**Domain A** **Images Count**	**Domain B** **Images Count**	**Image size**
Apple to Orange	995	1019	(256,256)
Monet to Picture	1072	6287	(256,256)
Summer to Winter	1231	962	(256,256)

6.5.2 Evaluation Metrics

In this study, we have utilized six evaluation metrics to analyze the proposals. Among these, three metrics are already used for fitness evaluation, while the rest three are used

to analyze and compare the model performance. The comparative performance analysis is done on these performance metrics as shown below:

1. SSIM (Structural Similarity Index): In this work, models with different settings are compared based on this metric [279]; however, other studies used it for image quality evaluation.

$$S_{sim}(x, y) = \frac{2\mu_x\mu_y + Q_1}{\mu_x^2 + \mu_y^2 + Q1} + \frac{2\sigma_{xy} + Q_2}{\sigma_x^2 + \sigma_y^2 + Q_2} \tag{6.12}$$

$$\begin{aligned} L_{sim}(G_a, G_b) = (1 - S_{sim}(G_a(x), x)) \\ + (1 - S_{sim}(G_b(y), y)) \end{aligned} \tag{6.13}$$

where μ_x, σ_x presents the mean and corresponding standard deviation (SD) of a fixed window centered like a pixel in x coordinate and similarly μ_y, σ_y in y coordinate. Q_1, Q_2 denote regularization term.

2. UQI (Universal Quality Index): The distortion effect in the final image is computed by UQI, with a focus on contrast distortion, luminance distortion, and loss of correlation [286].

$$UQI = \frac{4\mu_x\mu_y\sigma_{xy}}{(\sigma_x^2 + \sigma_y^2)(\mu_x^2 + \mu_y^2)} \tag{6.14}$$

where μ_x, μ_y represent the mean of fake and real image distribution respectively, and accordingly σ_x, σ_y are the corresponding SD.

3. MSE (Mean Squared Error): This metric is not specific for domain conversion purposes, but this is a nice metric to get insights into our model's performance.

I^{ref} and I^{test} are reference and test image respectively with size $N_1 \times N_2$ which are used in computing MSE as given below:

$$MSE(I^{ref}, I^{test}) = \frac{1}{N_1 \times N_2} \sum_{i=1}^{N_1} \sum_{J=1}^{N_2} (I_{ij}^{ref} - I_{ij}^{test})^2 \tag{6.15}$$

6.5.3 Effect of Pretrained Discriminator

Before proposing our first model EMOCGAN, we experimentally verified the reason for the proposal. The impact of considering a pre-trained discriminator on performance has been analyzed. In making the first effort, the discriminator from a trained Cyclic-GAN model has been utilized as an environment in EMOCGAN. In this version, the model's population sample only contains two generators. The results from this system were pretty bad, and the model was neither converging nor performing, which was further verified over different epochs. The sample images generated through this training are shown in Figure 6.7.

The reason behind the poor generation is as we already trained a Cyclic-GAN and the discriminator we took is very bad in classifying the images as real and fake thus, using that as a training environment cannot give a good model. Therefore, we added these two discriminators model to the population too and making the population sample now consisting of full Cyclic-GAN architecture.

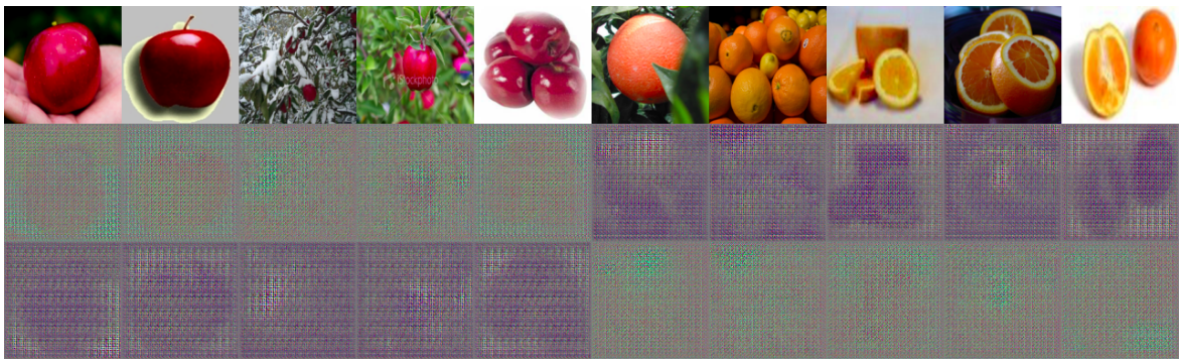


Figure 6.7: Sample images generated from the framework with pretrained discriminator

6.5.4 Hyperparameter Tuning

To select an optimal number of models for the next generation from the population of models, we experimented with different selection values and validated them in both directions based on SSIM. Both directions imply forward pass (F_Pass) that denotes image translation from domain ($a \rightarrow b$), whereas backward pass (B_Pass) denotes image translation from domain ($b \rightarrow a$).

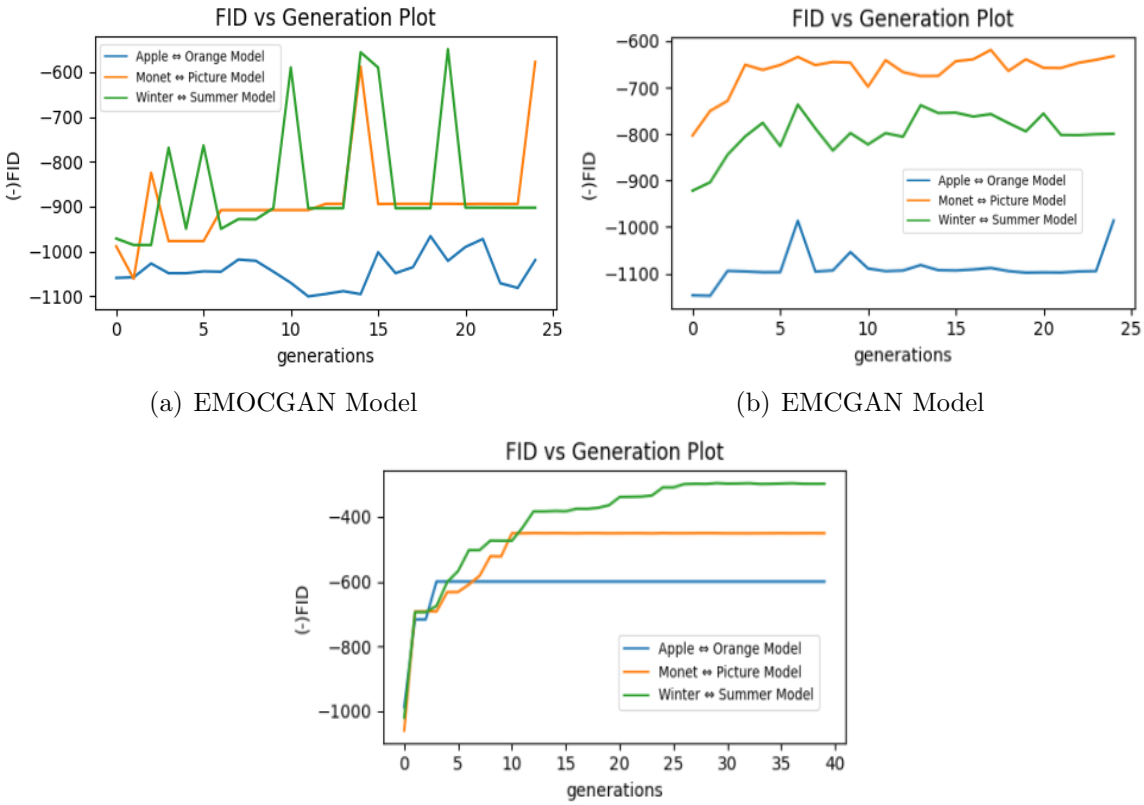
These results presented in Table 6.2 suggest that the selection count be set to 6 in order to give the model the best score while keeping the other configuration the same. It also shows that we can increase the selection per generation to get a better SSIM, but the model's performance suffers from an increase in this hyperparameter. Therefore, we keep the selection per generation to be 5 to achieve a good balance of performance and accuracy. The different parameters utilized for the proposal are shown in Table 6.3. These parameters are obtained by testing the model on the benchmark data sets we discussed. For the final model, we increased the number of generations and reduced the number of mutation iterations accordingly. The model has been trained on the

Table 6.2: Quantitative analysis of model with different hyperparameter

Final-Model with distinct hyperparameters	**SSIM value in F_Pass**	**SSIM value in B_Pass**
Generations : 50 Mutation iterations : 15 Selection per generation : 2	0.41	0.40
Generations : 50 Mutation iterations : 15 Selection per generation : 5	0.59	0.51
Generations : 100 Mutation iterations : 15 Selection per generation : 6	0.62	0.52

Table 6.3: Proposed model hyperparameters

Hyperparameters	Value
Number of Generations	30
Number of mutation Iterations	15
Number of selection models	5
Start Population	3
Learning rate for generator and discriminator	0.0002
Max number of models in any generation	14
Max number of randomly selected models	2
Ratio of evaluation and mutation data	0.1
Train Limit	300



(a) EMOCGAN Model

(b) EMCGAN Model

(c) QEMCGAN Model

Figure 6.8: Worst FID score plots for three real benchmark image datasets using proposed variants

datasets described in Table 6.1, and the results are discussed in the following section.

The score of the worst FID in every generation from the population of models in a different configuration is shown in Figure 6.8. These plots show that even the worst model in our population is improving, and the dips in the curves are due to the crossover operator and random selection, which puts even the models with lower scores in the next generation, but despite this, the FID value appears to be improving over time. The FID shown in Figure 6.8 is the negative of the actual FID value. It has been observed that the proposed QEMCGAN and EMCGAN generated more stable and better quality images as compared to the Cyclic-GAN as well as the first proposal EMOCGAN.

6.5.5 Comparison of Proposals using Benchmarks

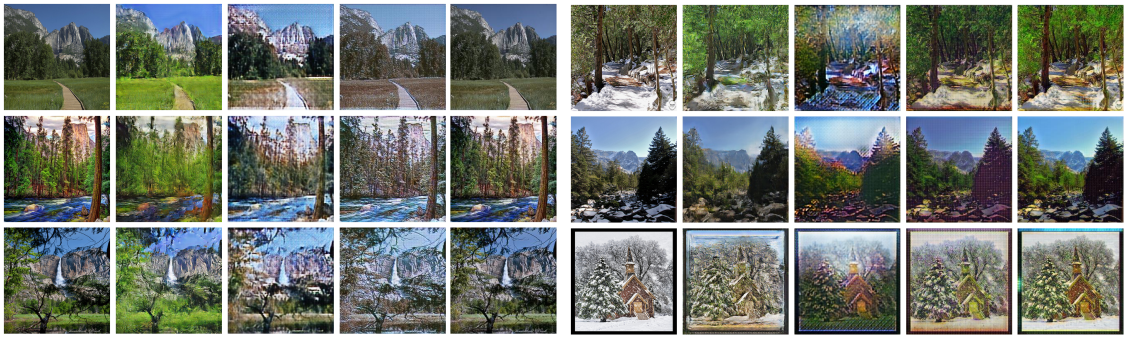
We have presented a qualitative and quantitative comparative analysis of the proposed models over our first proposal framework EMOCGAN. Figure 6.9 compares the EMOCGAN, EMCGAN model to the quantized version, QEMCGAN qualitatively, while quantitative results are shown in Table 6.4. The purpose of this in-depth model comparison is to justify the decision to incorporate the TVD and quantization into the EMOCGAN model. From Figure 6.9 (c) and 6.9 (d), it can be seen clearly that the fourth and fifth columns are capable of generating more realistic images while maintaining texture and contextual details more than the third column.

Table 6.4: Model evaluation

Datasets	**Metric**	**Cyclic-GAN**		**EMOCGAN**		**EMCGAN**		**QEMCGAN**	
		F_Pass	B_Pass	F_Pass	B_Pass	F_Pass	B_Pass	F_Pass	B_Pass
Apple and	SSIM	0.05466	0.04840	0.05603	0.05000	0.06647	0.05760	**0.07332**	**0.06146**
Orange	UQI	0.55305	0.57151	0.51040	0.52670	0.52690	0.53640	**0.55325**	**0.57174**
Winter and	SSIM	0.02584	0.03889	0.01703	0.02196	**0.02586**	**0.04352**	0.02495	0.03428
Summer	UQI	0.41560	0.46415	**0.45096**	0.43053	0.44144	**0.50786**	0.42659	0.48161
Monet and	SSIM	0.03881	0.02519	0.03361	0.02507	**0.04308**	**0.03276**	0.02365	0.03246
Pictures	UQI	0.47082	0.34683	0.47426	**0.43822**	**0.49040**	0.42009	0.46760	0.43012

(a) Apple to orange Conversion (b) Orange to apple Conversion

(c) Picture to Monet Conversion (d) Monet to Picture Conversion



(e) Summer to Winter Conversion (f) Winter to Summer Conversion

Figure 6.9: Comparison of QEMCGAN with the state-of-the-art. 1^{st} column shows the original image, 2^{nd} column shows result from Cyclic model, 3^{rd} column shows the results for EMOCGAN, 4^{th} column shows proposed EMCGAN and 5^{th} column is for proposed QEMCGAN Model.

A similar interpretation can be made from (Summer to Winter) and (Winter to Summer) conversion shown in Figure 6.9 (e) and (f). The observation made from the first row, and third row of (Summer to Winter), that EMCGAN (fourth column) and QEMCGAN (fifth column) are capable of removing greenery in the conversion far better than EMOCGAN (3rd column). In Orange to Apple, the third row shows

that proposals give favorable results while maintaining leaf structure and green color far better than others. Even UQI and SSIM are higher in the proposals as compared to EMOCGAN, with few exceptions. Overall, these images clearly demonstrate the QEMCGAN improved results. The quantization model outperforms the previous model in both size and performance, making it a significant improvement over the EMOCGAN model.

6.5.6 Comparison with State-of-the-Art Method

Comparative analysis with the state-of-the-art method (Cyclic-GAN) [272] is also performed, and the obtained inferences on three real-world image datasets are shown in Figure 6.9. EMOCGAN shows superior performance over Cyclic-GAN, and it is very reliable and consistent in producing visually realistic images while retaining background and salient objects. However, Cyclic-GAN misinterprets the original information during training. In the context of quantitative analysis, EMOCGAN attained the best SSIM value for a forward pass and backward pass (0.0560, 0.0500) for Apple to Orange, which is higher than Cyclic-GAN on the same data. It also attained higher UQI for the forward pass in the case of Winter to Summer and better UQI in both passes (0.4743, 0.4382), respectively, for Monet to Picture. There are a few exceptions where EMOCGAN has low UQI or comparable value against the state-of-the-art. Moreover, our other proposals, QEMCGAN and EMCGAN, are also good at retaining salient information aside from the concerned object, as evidenced by our results. Our model also brings the colors of the base object more accurately by not converting colors inappropriately, and it also considers the color variation that our object has in the corresponding environment. In some cases, our model outperforms the Cyclic-GAN model, and in some cases, it gives comparable results in terms of visual performance, as shown in Figure 6.9. QEMCGAN is also half in size as the EMCGAN and still be competing with a state-of-the-art model.

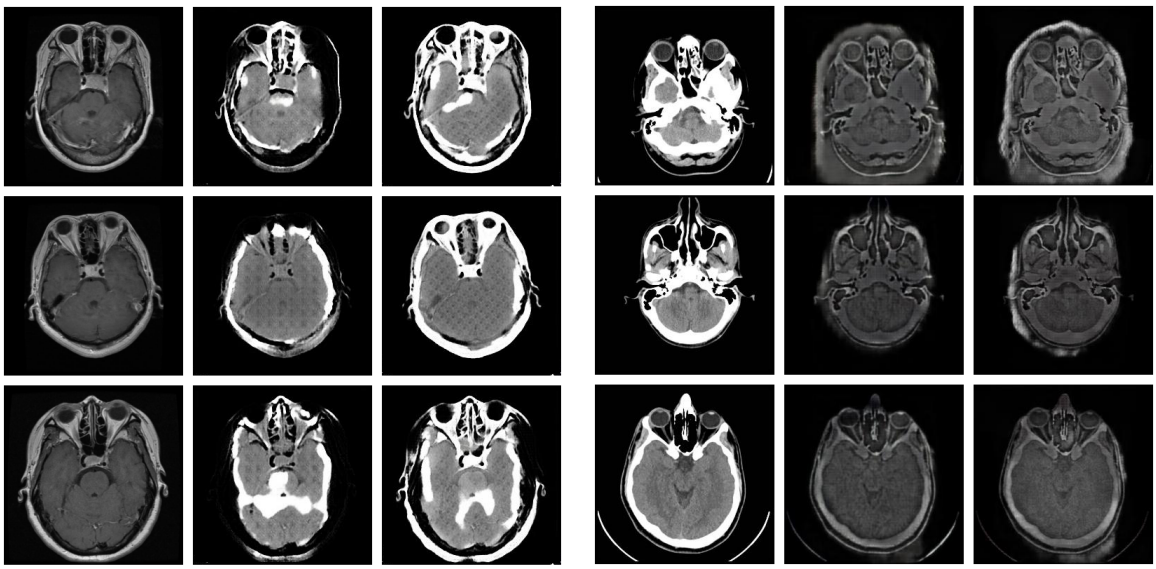
We also computed the SSIM Scores and the UQI for the presented images in Figure 6.9, along with the image distributions corresponding to them. SSIM is computed as the difference between the image produced and all of the other images in the distribution. The scores shown in Table 6.4 are the average of the complete set of generated images. Further, due to the lack of ground truth for unpaired I2I translation, we used the similarity of images with the distribution of the images instead of directly with the target images. Quantitative results show that the EMCGAN produces better SSIM and UQI than Cyclic-GAN and EMOCGAN in most of the cases, except for some exceptions. This quantitative analysis shows that the EMCGAN model is superior to all other models. Still, the QEMCGAN is very comparable to it in many places despite being half the size of the EMCGAN. We can conclude that the quantized model (QEMCGAN) is superior to the previous models.

6.5.7 Application to Medical Image Translation

In the previous section, we introduced a novel GAN architecture QEMCGAN and validated its efficacy on the benchmark datasets. Once it proved its efficiency, we applied it to the most challenging and demanding application-specific problem - "Medical image domain conversion" and described the model's use case. We have considered three healthcare image datasets.

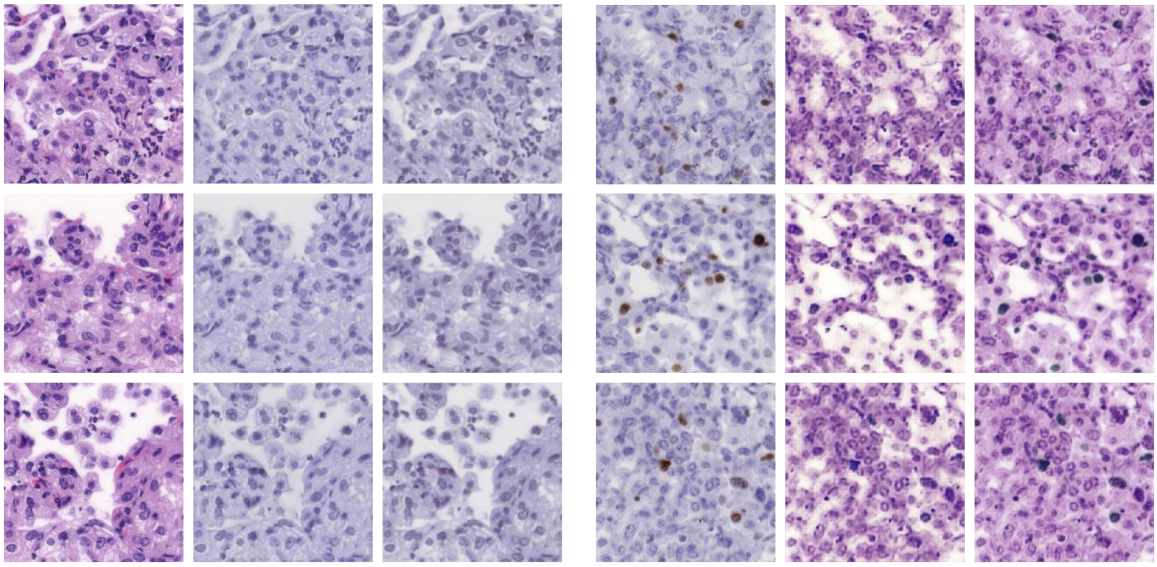
6.5.7.1 MRI to CT

The dataset used is the collection of unpaired MRI and CT images of the brain collected from different sources [287]. This dataset is created to use the domain conversion ability to obtain an MRI from a CT scan and vice versa. Both imaging modalities are non-invasive, so they are immensely popular in the clinical study of morphological plaque characteristics. Using this conversion, the model is aimed to reduce the diagnostic cost of a patient by giving two scans with the cost of one in real-time processing. As the



(a) MRI to CT conversion

(b) CT to MRI conversion



(c) H&E to Ki67 staining conversion

(d) Ki67 to H&E staining conversion

Figure 6.10: Sample results from proposed models. (1^{st} column shows the original image and 2^{nd} column shows the results for EMCGAN model and 3^{rd} column shows the result for proposed QEMCGAN model)

dataset was unpaired, there is no simple way to test the reliability of the model, but we have made the comparison of the two distributions produced by the model and the distribution we already have to justify the model's use. It is clearly visible from Figure 6.10 (a) and (b) that our proposed QEMCGAN produces a more visually realistic and

quality image while retaining the structure and contextual information as compared to previous work.

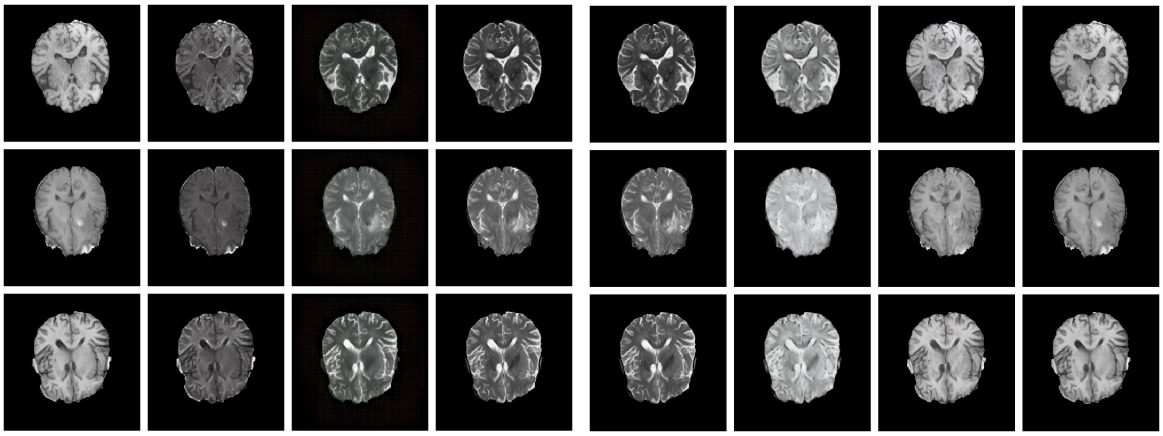
6.5.7.2 Virtual Restaining

The dataset used contains the images of lung lesion whole slide image with $H\&E$ (hematoxylin and eosin) and Ki67 (cellular marker used in immuno histochemistry-IHC) staining images [288]. This dataset is created from a very high-resolution image of the staining sample and then divided into patches of 256x256 pixels. Generally, $H\&E$ staining is common in digital pathology, but in some cases, it lacks providing enough contrast that makes it difficult to distinguish cancerous cells or non-cancerous cells. While Ki67 staining gives high-contrast images. IHC test confirms the malignancy type. It also lists key prognostic factors which direct the offered treatment to the patient. But it needs more labor and time, which makes it costly, and in a low-income country, only 1% of the cancer patient gets this examination [289]. Also, a lack of trained pathologists and functional equipment resulted in costly, unreliable quality, and delayed diagnosis.

So, we tried to automate this process through unpaired image translation for stain conversion, which seems to be a potential application for first opinions and clinical decisions. Generated images are shown in Figure 6.10(c) and (d). It shows that the structural details of original images are preserved due to considered structural similarity loss during translation. Qualitative results ensure that generated images can be used further for augmentation to overcome the overfitting problem in most of the deep architecture, as well as they can be used for the data distillation process to secure sensitive medical data on IoT or cloud-based applications.

6.5.7.3 MRI Scan Type Conversion

The dataset used for this conversion is created by extracting the T1(spin-lattice relaxation) and T2 type (spin-spin relaxation) MRI scan from the BraTs dataset [290].



(a) MRI-T1 to MRI-T2 scans (b) MRI-T2 to MRI-T1 scans

Figure 6.11: Sample results from our proposed model (1^{st} column shows the original image, 2^{nd} column shows the results for EMCGAN model, 3^{rd} column shows the results for QEMCGAN model and 4^{th} column shows the ground truth)

Both of the scans convey different information, and being able to get each other with the first scan can save a lot of money in any diagnostics. It is referred to widely for diagnostic and therapeutic purposes.

BraTs dataset is trimmed to contain 1000 images of each domain. Here, we have ground truth, so the metric computation is done using that. Qualitative results are shown in Figure 6.11 (a) and (b) and are capable of producing visually appealing and informative images.

6.5.7.4 Quantitative Analysis:

We have presented quantitative results using SSIM, UQI, and MSE on all three medical datasets in Table 6.5. Due to the lack of ground truth, here baseline is obtained by comparing the original image in the same data set with the image distribution to make an unbiased comparison. In terms of UQI, QEMCGAN outperform on $MRI \rightarrow CT$ and $H\&E \rightarrow Ki67$, while achieving higher SSIM value 0.5545 in $MRI \rightarrow CT$ conversion and in other cases ($MRIT2 \rightarrow T1$, Ki67$\rightarrow H\&E$) it gives competitive value even with low size model and beat baseline approach. Similar inferences can be made from other metrics shown in Table 6.5, which prove that QEMCGAN, which is almost half the

Table 6.5: Model evaluation for medical image translation

Datasets	Metrics	EMCGAN	QEMCGAN	Baseline
MRI → CT	SSIM	0.4370802	**0.554565**	0.470727
	UQI	0.643653	**0.67298**	0.620353
	MSE	**0.428117**	0.554565	0.47072
CT → MRI	SSIM	**0.436957**	0.120993	0.14062
	UQI	**0.803172**	0.57515	0.63524
	MSE	**0.112795**	0.12099	0.14062
HE → Ki67	SSIM	0.170070	**0.217548**	0.217346
	UQI	0.568930	**0.597732**	0.580972
	MSE	0.259044	0.162500	**0.156224**
Ki67 → HE	SSIM	0.132478	**0.139711**	0.139529
	UQI	0.558345	**0.574675**	0.574247
	MSE	0.308274	**0.306953**	0.324875
MRI T1 → T2	SSIM	**0.685811**	0.490034	0.680242
	UQI	**0.839224**	0.784692	0.783983
	MSE	0.085423	**0.076157**	0.079924
MRI T2 → T1	SSIM	**0.674365**	0.666019	0.661933
	UQI	**0.820500**	0.784193	0.783230
	MSE	0.1185287	**0.124180**	0.129696

size of EMCGAN, is generating more convincing and more true to distribution and thus make it a worthy and potential framework for medical image translations at the low-cost device.

6.6 Summary

Many issues can arise when training a GAN model, including mode collapse, local optima stagnation, and training difficulty. Thus, this study proposed three novel variants of multi-objective Cyclic-GAN, namely EMOCGAN, EMCGAN, and QEMCGAN, to address these issues. Through extensive experiments, it was proven that the proposed

models address the majority of these issues with vanilla GANs and Cyclic-GANs. The final proposal was also capable of generating more diverse, stable, visually realistic, and consistent images, including real-world natural images and medical data. These visually appealing results were obtained with less data, which is an important aspect of training GAN, demonstrating its efficiency and validating its acceptance for various applications. The advantages of Cyclic-GANs, evolutionary algorithms, three different well-known evaluation metrics, and multiple losses were integrated into our model to produce consistent and quality images, resulting in a generalized model. The model was capable of performing the majority of I2I translation tasks with a single training run. It demonstrated its reliability in reproducing the results shown in this study owing to the several randomizations we added to the model training framework. We also attempted to improve the robustness, efficiency in space, and complexity of EMCGAN while maintaining favorable performance compared to the state-of-the-art technique. Furthermore, there was no need to fine-tune the hyperparameters since the framework only required simple intuition to choose them. The fact that the model is quantized and ultimately produces a TFlite model qualifies it for a wide range of IoT applications.