

國立屏東大學
智慧機器人學系
Department of Intelligent Robotics

112學年度

專題研究期末報告

不同生成模型應用於圖像翻譯任
務之比較與探討

組 員：何杰懋 (CBD109019)
謝昕諺 (CBD109023)
莊佩蓁 (CBD109031)
陳冠霖 (CBD109035)

指導老師：楊柏遠 博士

中 華 民 國 112 年 12 月

摘要

人工智能與深度學習已然是目前科技發展的趨勢。在這之中，生成式 AI 的發展日新月異，從圖像生成、影片生成、聲音訊號生成、到聊天機器人等，都是各種不同的生成式 AI 所延伸出來的應用。生成式 AI 中又以圖像生成為最廣泛發展的領域。本專題研究分別應用不同的生成模型在圖像翻譯，並比較他們之間的效能差異；圖像翻譯是以一張圖片作為輸入，而後生成另一張不同風格的圖片。本研究使用生成對抗網路與擴散模型作為生成模型，並進一步地研究探討這些模型應用於圖像翻譯的能力以及優缺點。我們的展示頁面在 <https://nptuir.github.io/>。

關鍵字：生成式 AI、圖像翻譯、生成對抗網路、擴散模型

Abstract

Artificial intelligence (AI) and deep learning are currently the prevailing trends in technological development. Among these, generative AI has rapidly advanced, with various applications such as image generation, video generation, sound signal generation, and chatbots. Image generation is the most extensively developed field in generative AI. This project employs different generative models for image translation and compares their performance differences. Image translation involves taking an image as an input and generating another image with a distinct style. This research employs Generative Adversarial Network (GAN), Variational Auto-Encoder (VAE), and Diffusion Models as generative models. The study further investigates the abilities and advantages and disadvantages of these models as they are applied to image translation. Our presentation page is available at <https://nptuir.github.io/>.

Keyword: generative AI, image translation, generative adversarial networks, diffusion model

目錄

摘要	I
ABSTRACT	I
目錄	II
圖目錄	IV
表目錄	VI
演算法目錄	VII
第1章 緒論	1
1.1 研究背景	1
1.2 研究動機	1
1.3 研究目的	1
1.4 研究範圍與架構	1
1.5 研究方法與流程	2
1.6 預期效益	2
第2章 文獻探討	3
第3章 研究方法	6
3.1 資料前處理	6
3.2 條件式變分自動編碼器 (CVAE)	7
3.3 條件式生成對抗網路 (CGAN)	10
3.3.1. 生成對抗網路 (GAN)	10
3.3.2. 條件式生成對抗網路 (CGAN)	12
3.4 PIX2PIX	14
3.5 CYCLE-GAN	17
3.6 去噪擴散隱式模型 (DENOISING DIFFUSION IMPLICIT MODELS, DDIM)	20
3.7 ADAM 優化器	26
3.8 模型評估指標	27
3.8.1 結構相似性指數	27
3.8.2 峰值訊噪比	28
3.8.3 Kernel Inception Distance (KID)	29
3.8.4 Fréchet Inception Distance (FID)	30
3.8.5 Learned Perceptual Image Patch Similarity (LPIPS)	30
第4章 研究成果	32

4.1	實驗結果	32
4.1.1	CVAE 生成結果探討	32
4.1.2	CGAN 生成結果探討	34
4.1.3	Pix2Pix 生成結果探討	37
4.1.4	Cycle-GAN 生成結果探討	42
4.1.5	DDIM 生成結果探討	46
4.2	模型生成結果比較	49
第5章	結論與後續研究建議	52
5.1	結論	52
5.2	後續研究建議	52
參考文獻	54	
誌謝	56	
附錄	57	
附錄1.	硬體設備	57
附錄2.	軟體環境	57
附錄3.	變分自動編碼器 (VAE)之原理推導	57
附錄3-1.	VAE 訓練之 KL 散度計算	57
附錄3-2.	VAE 訓練之目標函數推導	60
附錄4.	擴散模型之原理推導	61
附錄4-1.	推導前向擴散之原始圖片與雜訊圖片之關係	61
附錄4-2.	推導目標函數	62
附錄4-3.	推導 $q(x_{t-1} x_t, x_0)$ 之分布平均與變異數	64
附錄4-4.	推導擴散模型之損失函數	64

圖目錄

圖 1. 研究流程圖	6
圖 2. 資料集內容，左邊為草圖標籤；右邊為真實圖片。	7
圖 3. CVAE 模型生成架構示意圖	8
圖 4. GAN 訓練優化目標示意圖	11
圖 5. Sigmoid 函數圖	11
圖 6. CGAN 模型訓練流程圖	12
圖 7. Pix2Pix 模型訓練流程圖	15
圖 8. Cycle-GAN 循環生成之概念圖	18
圖 9. Cycle-GAN 訓練與損失計算示意圖	20
圖 10. 對資料進行前向擴散之示意圖	21
圖 11. DDIM 採樣之示意圖	22
圖 12. DDIM 模型訓練流程	23
圖 13. 本研究建立之 CVAE 對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第50次訓練, (c) 第200次訓練, (d) 第1000次訓練, (e) 第5000次訓練, (f) 第9000次訓練。	33
圖 14. CVAE 訓練之損失變化	34
圖 15. CVAE 對於測試資料集的生成能力結果	34
圖 16. 本研究建立之 CGAN 對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第200次訓練, (c) 第1000次訓練, (d) 第5000次訓練, (e) 第10000次訓練, (f) 第30000次訓練。	36
圖 17. CGAN 訓練之對抗損失	37
圖 18. CGAN 對於測試資料之生成能力預測圖	37
圖 19. 原始論文之 Pix2Pix 對於訓練資料之訓練情形 (Bce+L1), (a) 第1次訓練, (b) 第1000次訓練, (c) 第10000次訓練, (d) 第25000次訓練, (e) 第50000次訓練, (f) 第150000次訓練。	39
圖 20. 本研究修改之 Pix2Pix 對於訓練資料之訓練情形 (L2+L1), (a) 第1次訓練, (b) 第200次訓練, (c) 第1000次訓練, (d) 第2500次訓練, (e) 第10000次訓練, (f) 第20000次訓練。	40

圖 21. 原始論文之 Pix2Pix 訓練損失, (a) 對抗損失, (b) 像素損失	41
圖 22. 本研究修改之 Pix2Pix 訓練損失, (a) 對抗損失, (b) 像素損失	41
圖 23. Pix2Pix 對於測試資料之生成能力比較, (a) 生成器損失使用 BCE+L1, (b) 生成器損失使用 L2+L1。	42
圖 24. 本研究建立之 Cycle-GAN 對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第200次訓練, (c) 第1000次訓練, (d) 第5000次訓練, (e) 第20000次訓練, (f) 第80000次訓練。	44
圖 25. Cycle-GAN 訓練損失, (a) 對抗損失, (b) 循環一致性損失, (c) 特徵損失。	45
圖 26. Cycle-GAN 對於測試資料之生成能力	46
圖 27. 本研究 DDIM 在 256×256 解析度下對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第5次訓練, (c) 第15次訓練, (d) 第25次訓練, (e) 第35次訓練, (f) 第50次訓練。	47
圖 28. 本研究 DDIM 在 64×64 解析度下對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第5次訓練, (c) 第15次訓練, (d) 第25次訓練, (e) 第35次訓練, (f) 第50次訓練。	48
圖 29. 本研究 DDIM 在不同解析度訓練下之訓練損失變化, (a) 256×256 解析度, (b) 64×64 解析度。	49
圖 30. DDIM 對於不同解析度下測試資料之生成能力比較, (a) 256×256 解析度, (b) 64×64 解析度。	49
圖 31. 對於不同草圖輸入到不同模型之生成圖片比較.....	50

表目錄

表 1. VAE 之網路架構.....	8
表 2. VAE 網路架構代號對照.....	9
表 3. CVAE 之參數設定表	10
表 4.CGAN 之網路架構	13
表 5. CGAN 網路架構代號對照	13
表 6. CGAN 之訓練超參數設定	13
表 7. Pix2Pix 之網路架構	16
表 8. Pix2Pix 網路架構代號對照	16
表 9. Pix2Pix 之訓練超參數設定	17
表 10. Cycle-GAN 之網路架構	18
表 11. Cycle-GAN 網路架構代號對照	18
表 12. Cycle-GAN 之訓練超參數設定	20
表 13. DDIM 之網路架構	24
表 14. DDIM 網路架構代號對照	25
表 15. DDIM 之訓練超參數設定	25
表 16. AlexNet 之網路架構	31
表 17. AlexNet 網路架構代號對照	31
表 18. CVAE 訓練結果之評估分數	32
表 19. CGAN 訓練結果之評估分數	35
表 20. Pix2Pix 訓練結果之評估分數	38
表 21. Cycle-GAN 訓練結果之評估分數	42
表 22. 各生成模型訓練之評估分數.....	51
表 23. 研究使用之硬體設備資訊.....	57
表 24. 研究使用之套件詳細資訊.....	57

演算法目錄

Algorithm 1: Transforming the latent vector distribution to a normal distribution (Custom layer)	8
---	---

第1章 緒論

1.1 研究背景

生成式 AI 的發展速度飛快，且在人工智慧領域中非常具有潛力。雖然生成式 AI 是從2014年生成對抗網路被提出後才得到重視，但這幾年研究發展卻非常迅速。現階段各種 AI 生成的圖像充斥在生活周遭，另外因應人類不同任務需求，各種多樣化的生成模型被提出。也有基於各種理論而被研究出來的各種生成模型架構，例如使用編碼器與解碼器為主的變分自動編碼器、以兩個網路對抗訓練為主的生成對抗網路、基於非平衡熱力學中的擴散過程為基礎的擴散模型。這些模型的原理天差地遠，但都能生成出圖片來，對此，本研究旨在研究以這些模型為基礎的圖像生成模型，在圖像翻譯的任務中其模型的生成效能、生成品質差異等進行一個比較並分析不同模型間的優劣區間。

1.2 研究動機

圖像生成的模型在未來發展中勢必是一個重要的方向，本研究想對於不同種類的圖像生成模型進行研究，研究其理論以及建模方式，直到實際用於訓練資料的表現等進行完整的實驗，並從中學習到更深度的知識以及累積當今熱門研究領域之研究經驗。

1.3 研究目的

本研究希望透過比較不同的生成模型演算法去探討出對於圖像翻譯的生成任務，研究中會使用草圖資料輸入至模型中並生成真實圖片。接著探討哪些模型在生成上效果較優秀，哪些模型在圖像生成上能力較差。並透過完整的比較去分析出不同模型之間的架構、參數設定等造成的影響。

1.4 研究範圍與架構

本研究使用變分自動編碼器(Variational Autoencoder, VAE)、條件式生成對抗網路(Conditional Generative Adversarial Networks, CGAN)、Pix2Pix、

Cycle-GAN、去噪擴散隱式模型(Denoising Diffusion Implicit Models, DDIM) 模型去進行圖像翻譯的任務。資料集使用 façade labels→photo，這是一個將房屋外觀標籤化成一張草圖，以及對應的房屋外觀圖，整個資料集草圖與真實圖片各有400張圖片。

1.5 研究方法與流程

實驗中首先進行資料的預處理，因為資料只有400張圖片，故本研究對其做了資料增強，將資料量擴增為2400張。接著建立多種生成模型進行比較。最後使用不同指標對各生成模型所生成的圖片進行評估，並分析各模型的效能。使用的指標有峰值訊噪比(Peak Signal-to-Noise Ratio, PSNR)、結構相似性指數(Structural Similarity Index, SSIM)、Kernel Inception Distance (KID)、Fréchet Inception Distance (FID)、Learned Perceptual Image Patch Similarity (LPIPS)。

1.6 預期效益

本研究預計分析各種生成模型的模型效能，並從中挑出一個最適合用於圖像翻譯任務的模型，並統整所有模型在圖像翻譯任務中的優缺點。

第2章 文獻探討

自從深度學習的概念逐漸普及，人工智慧發展的速度也隨之提升許多。深度學習能夠達成的任務也非常廣泛，例如能夠使用深度學習來進行迴歸分析、分類預測、影像處理、資料生成、機器人控制[1]等。迴歸分析中，文獻[2]使用了多種迴歸的演算法來對現有的任務進行全面性的分析與概述。此外也有使用半監督式學習 (Semi-Supervised Learning, SSL)與交替學習 (Alternate Learning)來解決深度學習模型的參數估計問題[3]，該研究所提出的方法能夠緩解樣本量不足的問題，使研究人員不需要再收集其他具有目標值的額外樣本。在分類預測中也有許多研究，例如文獻[4]使用深度卷積網路來根據肺部超音波的照片來判斷 COVID-19 的分類與成像生物標記的定位等。

接著在影像處理中深度學習也能夠達成廣泛的任務，藉由卷積神經網路 (Convolutional Neural Network, CNN)[5]能夠更良好的處理圖片資料的特徵等訊息，也能將圖片透過深度卷積神經網路進行各種處理以萃取特徵等。除了對圖片進行分析，深度學習也可以使用 CNN 將圖片進行編碼，編碼成一個固定長度的向量，接著再進行解碼，根據編碼內容透過 CNN 再反轉換回原始的圖片。這個應用即為自動編碼器 (Auto-Encoder, AE)，這個網路的概念分為編碼器與解碼器。編碼器的任務是將圖片編碼為一定長度的隱向量；解碼器的任務是將隱向量反轉變回原始圖片。這個做法有許多用意，可以降低資料傳遞時所佔用的空間，即將圖片透過網路傳給其他用戶時在伺服端會將資料壓縮成隱向量，接著傳送給另一用戶時再進行解碼將圖片反轉回原始圖片。另外也有許多應用，例如[6]使用基於注意力機制的 AE，該研究所提出的方法是用於預測藥物與疾病關聯的應用，方法中整合了藥物與疾病的關聯、不同疾病的相似度、不同藥物的相似程度、藥物的屬性等資料，根據這些資料建構了一個圖卷積自動編碼器網路，用於學習每個藥物與疾病的拓撲表示，藉此可以判斷藥物與疾病的關聯。雖然 AE 的用途廣泛，也可以生成資料。但是因為隱向量的分布狀況無法計算出來，人類無法模擬這些隱向量分布，從此分布中採樣一段編碼來讓解碼器生成資料。所以將 AE 變成生成模型還需要克服隱向量編碼的問題。

在2013年，變分自動編碼器 (Variational Auto-Encoder, VAE)[7]被提出來，這項研究增加了編碼器編碼的限制，使其編碼的結果需要符合常態分布。如此

一來就可以從常態分佈中取樣一段編碼用於生成資料。自此之後自動編碼器能夠憑空生成各種圖片，是最基本的生成式 AI 應用。接著在2014年生成對抗網路 (Generative Adversarial Network, GAN)[8]被提出，這個生成模型架構使用兩個神經網路進行訓練，分別為生成器 (Generator)與判別器 (Discriminator)，生成器與判別器會相互對抗式的訓練，生成器訓練目標是根據常態分佈中採樣出來的雜訊來生成圖片，而判別器的訓練目標是判斷輸入圖片是來自資料集的真實圖片抑或是來自於生成器的假圖片。透過不斷的對抗訓練，生成器生成的圖片會越來越接近真實圖片；判別器的判斷能力也會越來越強大。訓練到最後達到納許均衡時，生成器與判別器的能力就不會再因訓練而進步，此時生成對抗網路模型即訓練完畢。在 GAN 被提出後過沒多久深度卷積生成對抗網路 (Deep Convolutional Generative Adversarial Networks, DCGAN)[9]也被提出，此模型使用了 CNN 來代替原始 GAN 使用全連接層的應用，使生成圖片的品質變得更好，目前許多 GAN 的應用都是使用 CNN 為主架構。此外在 GAN 提出後的幾年中，關於 GAN 的變種與應用研究呈現指數型的增長，代表著 GAN 在生成式 AI 領域的潛力無窮。在能夠生成圖片後學術界開始致力於發展條件式的生成，使 GAN 的生成結果能夠被人類控制。在此之下條件式生成對抗網路 (Conditional Generative Adversarial Network, CGAN)[10]也在2014年被提出。此模型將條件向量與雜訊一同被輸入至生成器中，藉由條件遷入使生成結果變的可控；另外判別器也要訓練圖片的真假以及圖片是否有照條件生成。在 GAN 的發展大致確認後研究人員開始將不同架構、不同目標函數、不同預訓練模型等加入 GAN，希望以此能夠更符合的生成人類要求的圖片。在2016年，基於 CGAN 的模型 Pix2Pix 被提出[11]，與 CGAN 不同的是 Pix2Pix 直接將圖片輸入，藉此輸出不同風格的圖片。除此之外 Pix2Pix 還在生成器中使用 U-Net 架構使生成器能夠更細緻的處理圖片特徵的細節，此外在判別器中使用 PatchGAN 的架構讓判別器不再是對一張圖片判斷真假而是對部分圖片判斷真假，讓模型能夠知道生成圖片哪個部分還不夠逼真。在 Pix2Pix 被提出之後，2017年同一作者又提出了 Cycle-GAN[12]，這個模型改良了 Pix2Pix 只能使用成對數據來訓練的缺點，Cycle-GAN 使用兩組生成器與判別器分別訓練其風格變化的特徵，在目標函數上使用了迴圈一致性的概念使圖片在經過兩次風格轉換後能夠再次變回原圖。此外在模型架構上除了使用 U-Net 以外還使用了殘差的概念，使模型

更深層卻又可以避免因網路層太深而導致梯度消失。透過這些概念 Cycle-GAN 能夠更良好的達成風格轉換的任務，生成的圖片品質又更加優秀，並且在資料集的準備上能夠更有彈性。在2020年，擴散模型被提出，其中最原始的去噪擴散機率模型 (Denoising Diffusion Probabilistic Models, DDPM)[13]是目前擴散模型的始祖，他透過擴散時間為圖片加上噪音，使圖片在經過一定擴散時間後完全變為一張雜訊圖片。接著訓練神經網路來模擬前向擴散中加上雜訊的平均期望分布，根據此分布生成逆向擴散的雜訊分布並應用於雜訊去噪，經過一定時間的去噪後雜訊圖片即會被去噪變回原始圖片。該項研究的生成能力甚至比 GAN 更好也更穩定，所以目前生成模型大多數都使用擴散模型來生成圖片。不過在模型擴散過程中的採樣時間造成模型訓練相當久、計算量非常龐大，於是有一些研究探討如何更高效的採樣、擴散。2020年底去噪擴散隱式模型 (Denoising Diffusion Implicit Models, DDIM)[14]被提出。與 DDPM 不同，DDIM 將採樣過程不再限制於馬可夫鏈，使採樣速度提高。接著計算出逆向擴散的隱式解，使生成結果變的固定，DDIM 的改良對於加速訓練有著非常大的幫助。

本研究使用 CGAN、VAE、Pix2Pix、Cycle-GAN、DDIM 來進行圖片的風格變換任務，將圖片草圖經過模型計算後生成解析度較高的圖片，並使用多種訓練指標來評估生成結果。並研究目前模型在訓練資料集的表現與限制，透過全面的分析來探討模型的效能與優缺點。

第3章 研究方法

本研究旨在使用不同模型來進行影像翻譯之風格轉換任務，首先先對資料集進行處理、分析。接著分別建立五種不同的模型，這些模型分別有條件式變分自動編碼器 (Conditional Variational Auto-Encoder, CVAE)、條件式生成對抗網路 (Conditional Generative Adversarial Network, CGAN)、去噪擴散隱式模型 (Denoise Diffusion Implicit Model, DDIM)、Cycle-GAN、Pix2Pix。其中 DDIM 因為資料集數量不足以進行完整的訓練，故本研究中僅討論 DDIM 的生成過程等，並無影像翻譯風格變換之應用。模型訓練時也會探討訓練設定所造成的影响，並在訓練完成之後使用多種模型評估指標來判斷模型生成的優劣。接著再進一步探討各種模型及不同訓練設定下訓練成果的效能。研究整體的流程步驟如圖 1 所示。

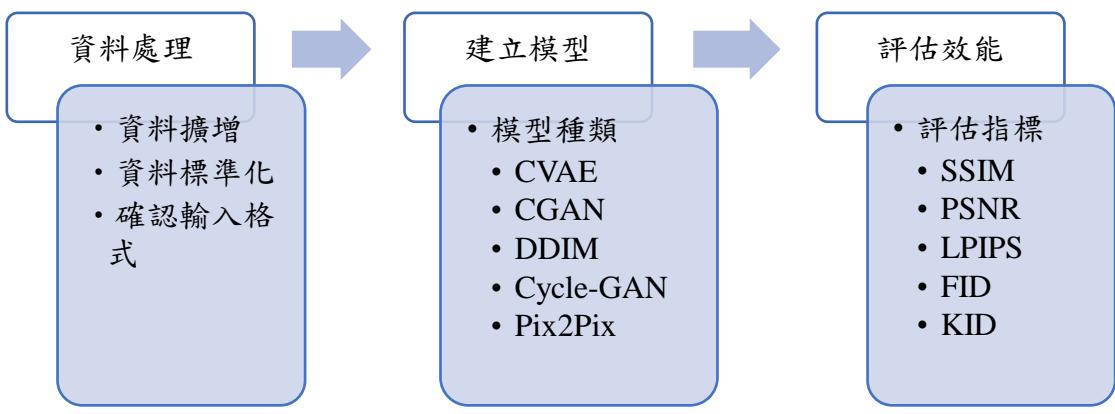


圖 1. 研究流程圖

3.1 資料前處理

本研究使用 façade labels→photo [15] 資料集作為訓練資料，這個資料集包含一張草圖與一張真實圖片。資料集中的圖片如圖 2 所示，可以看到左邊為草圖，右邊為對應草圖的房屋圖片，此資料集為了也能拿來訓練圖像修復能力而在資料中的一些真實圖片會添加隨機遮蔽。

Facades Dataset Preview

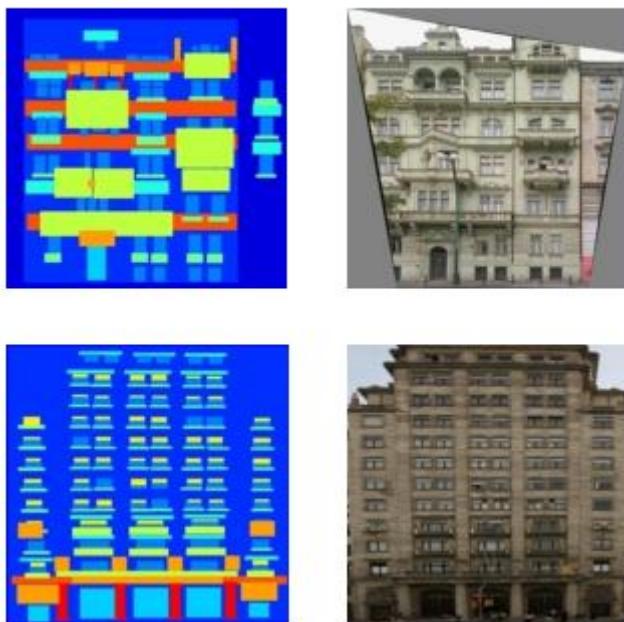


圖 2. 資料集內容，左邊為草圖標籤；右邊為真實圖片。

實驗中使用了資料增強，將圖片放大到 300×300 的解析度，再隨機從圖片中取 256×256 的區塊作為新圖片、以及將圖片進行鏡像水平翻轉等。將資料擴充到2400張圖片。

3.2 條件式變分自動編碼器 (CVAE)

自動編碼器(Auto-Encoder, AE) [16]是一種深度學習模型，通常用於無監督學習和降維。它的主要目標是學習如何有效地表示輸入數據，同時也能用於數據壓縮和去噪。變分自編碼器(Variational Auto-Encoder, VAE) [7]則是自動編碼器的一種變體，它的編碼器所輸出的數據則是遵從高斯分布，是一種機率模型。

條件變分自編碼器(Conditional Variational Auto-Encoder, CVAE)基本結構為編碼器(Encoder)和解碼器(Decoder)。模型架構如表 1，網路架構代號對照如表 2，編碼器的功能是將輸入數據轉換為潛在變數的機率分布，而在編碼器的最後一層為自定義層，負責處理潛在向量空間的部分，計算方式如Algorithm 1。解碼器則是接收由編碼器生成的潛在變數樣本，然後嘗試生成與原始輸入數據相似的數據，使其可以用來生成新的數據樣本。本研究

CVAE 的模型架構生成的示意圖如圖 3。

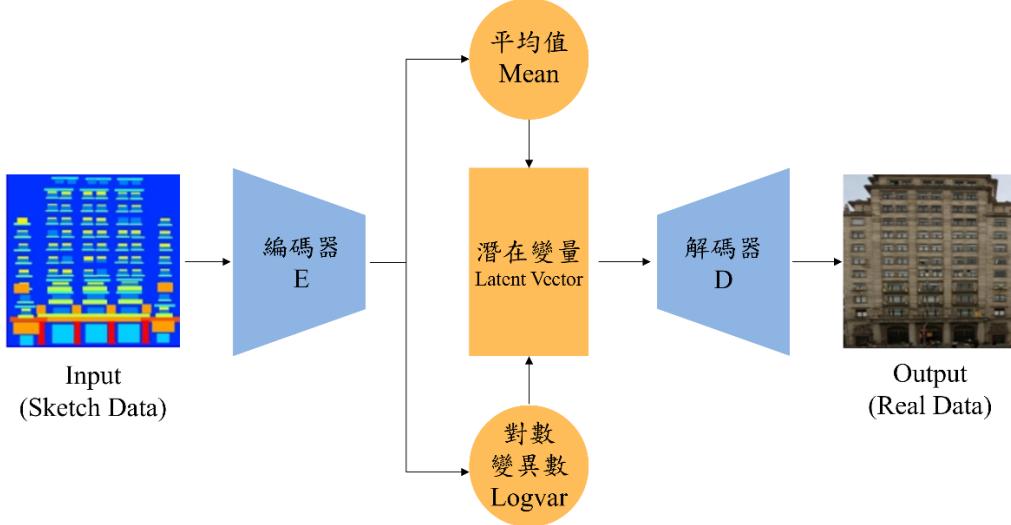


圖 3. CVAE 模型生成架構示意圖

Algorithm 1: Transforming the latent vector distribution to a normal distribution (Custom layer)

Input: Encoder output v that the dimension is 2048 ($2d$);

Output: The Decoder's input (z) that dimension is 1024 (d);

1. Split the v into $mean$ and $logvar$ vectors of length d ;

2. Sample the ε of length d from $\mathcal{N}(0,1)$;

3. Calculate the z using (4), the dimension of z is 1024 (d);

表 1. VAE 之網路架構

網路名稱	層數	網路層架構	輸出形狀
編碼器 <i>E</i>	1	Encoder Input	(256, 256, 3)
	2	C32-L	(128, 128, 32)
	3	C64-L	(64, 64, 64)
	4	C128-L	(32, 32, 128)
	5	C128-L	(16, 16, 128)
	6	Flatten	(32768,)
	7	Fully Connected 2048	(2048,)
	8	Custom Layer	(1024,)
解碼器 <i>D</i>	1	Decoder input	(1024,)
	2	Fully Connected 32768	(32768,)
	3	Reshape	(16, 16, 128)
	4	CT128-L	(32, 32, 128)
	5	CT128-L	(64, 64, 128)
	6	CT64-L	(128, 128, 64)
	7	CT32-L	(256, 256, 32)
	8	Output	(256, 256, 3)

表 2. VAE 網路架構代號對照

代號名稱	對應意思
Cn	具有 n 個神經元的卷積層
L	LeakyReLU 激活函數層
CTn	具有 n 個神經元的反卷積層
R	ReLU 激活函數層
Custom Layer	自定義網路層

CVAE 的損失函數(1)分為兩個部分，第一個是重構損失(2)，以及 KL 損失(3)。重構損失表示輸入數據的重構誤差，即模型輸出的數據與原始數據在每個像素值上的差異，通常使用 L2 誤差來進行衡量。重構損失中的 h 為圖片高、 w 為圖片寬；而 KL 損失又稱作相對熵，它衡量的是相同事件空間裡的兩個機率分佈的差異情況，公式中的 k 是潛在空間向量的維度， μ 和 σ 分別是潛在空間向量的平均值和標準差，透過公式可以計算出編碼器生成的平均值與變異數與平均為 0、變異數為 1 的常態分佈 $\mathcal{N}(0,1)$ 的分布差異。這麼做的用意是希望編碼器可以生成符合要求的常態分佈編碼並用做於解碼，之後就可以從常態分佈中採樣一段向量用作於圖片的生成。常態分佈計算公式如(4)，其中 z 是其抽樣得到的潛在空間向量， ε 是從標準常態分佈 $\mathcal{N}(0,1)$ 抽樣得到。編碼器生成分布的平均值與變異數與 $\mathcal{N}(0,1)$ 之 KL 散度的詳細推導過程如附錄 3-1。

$$\mathcal{L}_{VAE} = \mathcal{L}_{Recon} + \mathcal{L}_{KL} \quad (1)$$

$$\mathcal{L}_{Recon} = \sum_{i=1}^h \sum_{j=1}^w \sum_{k=1}^3 (y_{i,j,k} - CVAE(x)_{i,j,k})^2 \quad (2)$$

$$\mathcal{L}_{KL} = -0.5 \sum (1 + \log(\sigma^2) - \mu^2 - \sigma^2) \quad (3)$$

$$z = \varepsilon \odot e^{0.5 \log var} + mean \quad (4)$$

本研究的 CVAE 模型訓練超參數設定如表 3，其中模型訓練 10000 次，每次訓練一批資料使用 64 對圖片，網路架構中的卷積核大小皆為 3，而解碼器輸出層的卷積步長為 1，其他卷積隱藏層的卷積步長皆為 2，優化器使用 Adam 優化器，學習率為 0.002， β_1 與 β_2 分別為 0.9 和 0.999。

表 3. CVAE 之參數設定表

參數名稱	參數值
訓練批次量	64
訓練次數	10000
潛在向量空間維度	1024
學習率	0.0001
Adam β_1	0.9
Adam β_2	0.999

3.3 條件式生成對抗網路 (CGAN)

3.3.1. 生成對抗網路 (GAN)

生成對抗網路(Generative Adversarial Network , GAN)由兩個網路構成，分別是判別器(Discriminator)與生成器(Generator)，生成器從潛在空間(latent space)中隨機取樣作為輸入，輸出結果需要盡量模仿訓練集的真實圖像。判別器的輸入則為真實圖像或生成器的輸出，目的是將生成器的輸出從真實圖像中盡可能分辨出來。而生成器則要盡可能地欺騙判別器。兩個網路相互對抗、不斷調整參數，最終目的是使判別器無法判斷生成器的輸出結果是否真實。

生成器訓練與判別器訓練都是使用交叉熵作為損失函數，不過生成器與判別器對目標函數的優化目標不同，因而形成對抗式的訓練。其中 y 是真實圖像， P_{data} 是真實數據分佈， $x \sim P_{data}$ 代表從真實數據中取出一批資料， $x \sim p_z(z)$ 是雜訊分佈， $z \sim p_z(z)$ 代表從常態分佈的雜訊中取出一批雜訊 z ， $D(x)$ 是判別器對 x 的真假判斷機率， $G(z)$ 是生成器根據雜訊生成的圖像。目標函數會衡量判別器對真實圖像和生成圖像的分類能力，當判別器固定且在訓練生成器時，生成器會嘗試最小化該函數;而當生成器固定、在訓練判別器時，判別器會嘗試最大化這個函數。在訓練過程中，判別器對生成圖像的判斷機率越接近1越好，判別器對真實圖像的判斷機率越接近0越好，生成圖像的判斷機率越接近0越好。

GAN 訓練的目標函數如(5)，其中生成器與判別器優化函數的目標不同。圖 4 中的 $\log(1 - D(G(z)))$ 為生成的假圖片經過判別器判斷的結果， $\log(D(x))$ 為真實圖片經過判別器判斷的結果。橫軸是判別器的判斷，而判別器是二元分類，此時會用 Sigmoid 激活函數(6)，函數圖如圖 5，值域為 $[0,1]$ 。縱軸是將判別結果經由目標函數計算出來的結果值。當($D(G(z))=1$)代

表生成器可以以假亂真，因為生成結果經過判別器函數結果為1，代表判別器認為生成圖為真實圖。 $\log(1 - D(G(z)))$ 在 $D(G(z))$ 等於1的位置時為最低值，即為生成器要最小化目標函數的意思；當($D(x)=0$)時 $\log(1 - D(G(z)))$ 在最大值，即判別器要最大化目標函數的意思。真實圖片為真($D(x)=1$)時 $\log(D(x))$ 也在最高點。

$$\min_G \max_D V_{GAN}(G, D) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))] \quad (5)$$

$$Sigmoid(x) = \frac{1}{1 + e^{-x}} \quad (6)$$

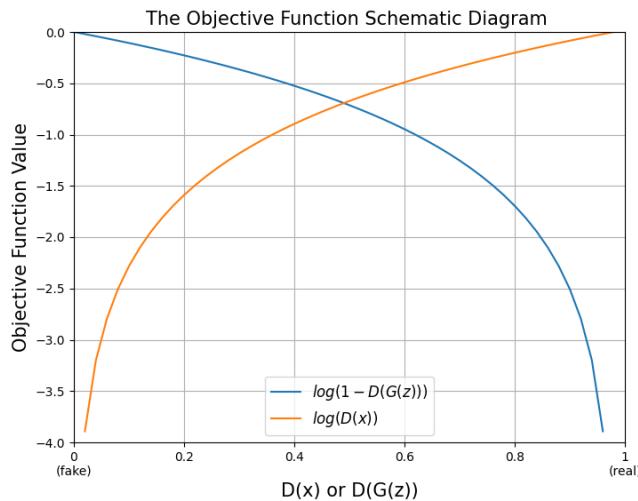


圖 4. GAN 訓練優化目標示意圖

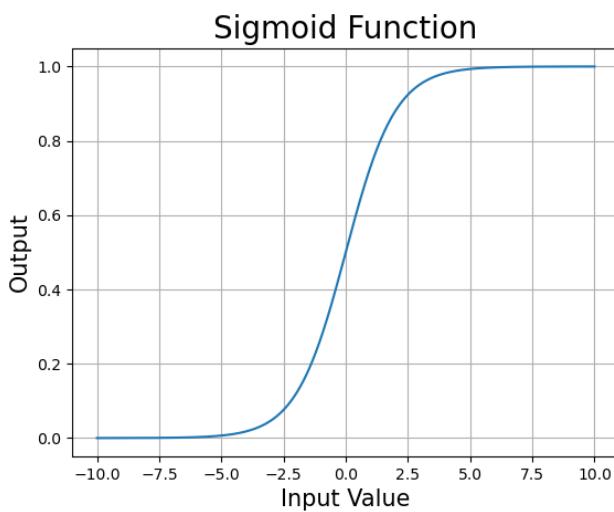


圖 5. Sigmoid 函數圖

3.3.2. 條件式生成對抗網路 (CGAN)

CGAN 的訓練目標函數(7)，其中有判別器(Discriminator, D)生成器(Generator, G)，與 GAN (5)的差異就是 CGAN 會使用條件輸入來引導生成，不像 GAN 生成的結果是人為不可控的。在 CGAN 裡，生成器 G 跟判別器 D 都以草圖 x 輸入作為條件， y 為真實房屋圖片。我們可以通過輸入 x 到生成器跟判別器中作為輸入層來達到訓練目的，在生成器中草圖作為資料輸入，而在判別器中， x 跟 y 會一起輸入做判斷。本研究 CGAN 的訓練流程如圖 6，基本上 GAN 的訓練流程都大同小異。模型的架構如表 4，其中各層的代號對照如表 5。CGAN 模型訓練超參數設定如表 6，其中訓練 30000 次，每次訓練一批資料使用 64 張圖片，優化器使用 Adam 優化器，生成器與判別器學習率皆為 0.0002， β_1 為 0.9 與 β_2 皆為 0.999。

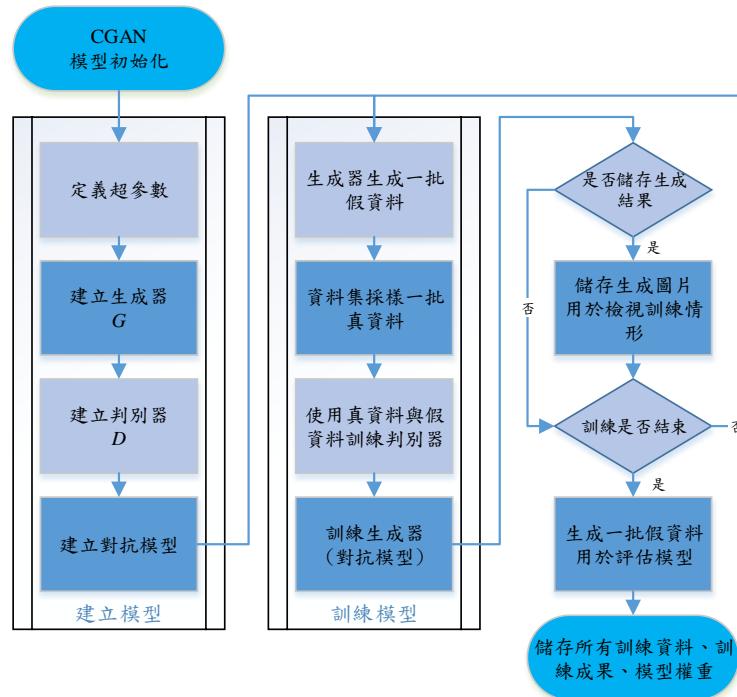


圖 6. CGAN 模型訓練流程圖

表 4.CGAN 之網路架構

網路名稱	層數	網路層架構	輸出形狀
生成器 <i>G</i>	1	Generator input	(256, 256, 3)
	2	FC3-BN-D45-R	(256, 256, 3)
	3	Reshape	(256, 256, 3)
	4	CT3S2-BN-R	(512, 512, 3)
	5	UpSampling Size=2	(1024, 1024, 3)
	6	CT2-BN-R	(1024, 1024, 2)
	7	CT1-BN-R	(1024, 1024, 1)
	8	C3S4-Sigmoid	(256, 256, 3)
判別器 <i>D</i>	1	Discriminator Sketch input	(256, 256, 3)
	2	C64S2-D40-R	(128, 128, 64)
	3	C128S2-D40-R	(64, 64, 128)
	4	C256S2-D40-R	(32, 32, 256)
	5	C512S1-D40-R	(32, 32, 512)
	6	Flatten	(524288,)
	7	FC1-Sigmoid	(1,)

表 5. CGAN 網路架構代號對照

代號名稱	對應意思
FCn	具有 <i>n</i> 個神經元的密集連接層
CnSm	具有 <i>n</i> 個神經元的卷積層，步進值為 <i>m</i>
UpSampling	上採樣層，Size為輸出特徵長寬放大的倍率
BN	批次正規化層
Sigmoid	Sigmoid激活函數層
CTnSm	具有 <i>n</i> 個神經元的反卷積層，步進值為 <i>m</i>
Dn	<i>n</i> %的丟棄層
R	ReLU激活函數層
Concat n & m	將 <i>n</i> 與 <i>m</i> 層輸出做合併，用於判別器

表 6. CGAN 之訓練超參數設定

參數名稱	參數值
訓練批次量	64
訓練次數	30000
生成器學習率	0.0002
生成器 β_1	0.9
生成器 β_2	0.999
判別器學習率	0.0002
判別器 β_1	0.9
判別器 β_2	0.999

$$\min_G \max_D V_{CGAN}(G, D) = \mathbb{E}_{y \sim p_{data}(y)} [\log D(y|x)] + \mathbb{E}_{x \sim p_{data}(x)} [\log (1 - D(G(y|x)))] \quad (7)$$

3.4 Pix2Pix

Pix2Pix [11]是2016年被提出的生成模型，他是基於 CGAN 而優化改良的模型。該模型只接受一對一的資料被送入並進行圖像翻譯的任務。此模型分為兩個網路，生成器與判別器，其模型架構如表 7。架構之層數代號如表 8所示，其中若沒特別說明卷積層與反卷積層的步進值，則步進值 m 為 2；LeakyReLU 的負數部分斜率為0.2；每個卷積層與反卷積層的卷積核大小都為 (4,4)。生成器使用 U-Net 結構，此結構類似於 AE 網路，只是在下採樣層與上採樣對應層中會進行跳接，使梯度能夠傳達到前面的網路，並使下採樣的特徵與上採樣的圖片生成能夠有對應的關係。判別器使用 PatchGAN 架構，此架構會將輸入圖片分成許多部分，並判斷這些部分的真假，並相加以此判斷整張圖片的真假。不同於以往的 GAN 對於一張完整圖片判斷真假，此作法的好處是能夠更兼顧到細節的判斷。

Pix2Pix 模型訓練之步驟如圖 7所示，首先定義所使用到的超參數等，並建立生成器與判別器，接著將生成器輸出與判別器輸入連接，以建立對抗模型。建立對抗模型時要將判別器的權重固定住，即只訓練生成器。這麼做的原因是因為要讓生成器盡可能生成可以騙過判別器的圖片，此時判別器的狀態就不可更新，否則在訓練生成器時也會訓練到判別器。在訓練模型時生成器首先會生成一批假資料，並從資料集中採樣一批資料供判別器訓練其判別能力。訓練完判別器後接著訓練生成器的生成能力，此時會使用對抗模型，讓生成器生成一批資料後給判別器判斷生成結果，若生成器生成結果讓判別器判斷為真實圖片則代表生成器有成功訓練。最後訓練一批資料之後會根據當前訓練步數決定是否儲存訓練資料，此步驟的用意是將訓練過程產生的圖片儲存起來以便後續分析訓練情形。接著判斷訓練是否結束，若還沒結束則返回訓練模型的步驟再進行訓練；訓練完成後則生成一批資料用於模型評估，以及儲存訓練的模型權重、訓練損失、訓練成果等資料，用於後續成果分析。

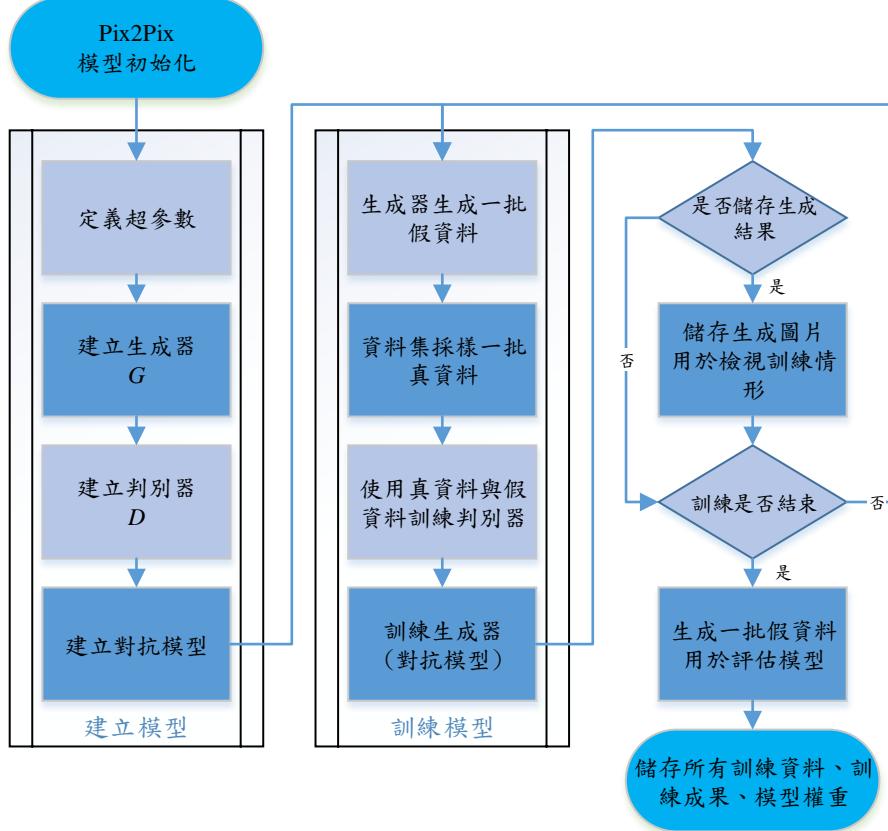


圖 7. Pix2Pix 模型訓練流程圖

Pix2Pix 在原始論文中的生成器學習的目標函數如 (8)式所述，其中分為兩個部分，第一個是原始 CGAN 的目標函數，以及另一個 L1損失 (11)，該損失是生成器會學習將風格轉換後的圖片與真實圖片進行逐個像素的比較，比較生成圖片之平均絕對誤差，此即希望生成器能夠根據輸入圖片生成一張對應且正確的圖片。公式中 w 代表圖片寬、 h 為圖片高、 y_{ijk} 為真實圖片在 (i, j, k) 座標的像素值、 y_{ijk} 會與生成圖片 $G(x)$ 在同樣座標下的像素值進行相減並取絕對值最後計算平均即為真實照片與生成照片之 L1損失。此損失會乘上一個超參數 λ ，本實驗中 λ 值設定為 100。另外在本實驗中有嘗試修改目標函數的第一個項次，修改過的方程式如 (9)，我們將判斷真假的交叉熵損失改成 L2損失，L2損失計算如 (10)，其中 f_w 與 f_h 是經過判別器計算之圖片特徵長寬。 $D(y)_{ij}$ 是在 (i,j) 位置中判別器針對真實資料 y 的區塊所進行的判斷值； $D(G(x))_{ij}$ 是在 (i,j) 位置中判別器針對生成資料 $G(x)$ 的區塊所進行的判斷值。此公式會計算判斷誤差的平方，並計算後續梯度以及模型優化。

本研究的 Pix2Pix 模型訓練超參數設定如表 9，其中使用本研究修改的目標函數設定上訓練 80000 次，每次訓練一批資料只使用 1 對圖片，優化器使用 Adam 優化器，生成器與判別器學習率皆為 0.0002。使用原始論文的目標函數設定上訓練 150000 次，每次訓練使用 2 對圖片，其餘設定與本實驗修改後的方法一致。

表 7. Pix2Pix 之網路架構

網路名稱	層數	網路層架構	輸出形狀
生成器 <i>G</i>	1	Generator input	(256, 256, 3)
	2	C64-L	(128, 128, 64)
	3	C128-IN-L	(64, 64, 128)
	4	C256-IN-L	(32, 32, 256)
	5	C512-IN-L	(16, 16, 512)
	6	C512-IN-L	(8, 8, 512)
	7	C512-IN-L	(4, 4, 512)
	8	C512-IN-L	(2, 2, 512)
	9	C512-IN-L	(1, 1, 512)
	10	C512-IN-L	(1, 1, 512)
	11	CT512S1-IN-D50-R, Concat 9	(1, 1, 1024)
	12	CT1024-IN-D50-R, Concat 8	(2, 2, 1536)
	13	CT1024-IN-D50-R, Concat 7	(4, 4, 1536)
	14	CT1024-IN-R, Concat 6	(8, 8, 1536)
	15	CT1024-IN-R, Concat 5	(16, 16, 1536)
	16	CT512-IN-R, Concat 4	(32, 32, 768)
	17	CT256-IN-R, Concat 3	(64, 64, 384)
	18	CT128-IN-R, Concat 2	(128, 128, 192)
	19	CT3-Tanh	(256, 256, 3)
判別器 <i>D</i>	1-1	Sketch data input	(256, 256, 3)
	1-2	Discriminator input	(256, 256, 3)
	2	Concat 1-1 & 1-2	(256, 256, 6)
	3	C64-L	(128, 128, 64)
	4	C128-IN-L	(64, 64, 128)
	5	C256-IN-L	(32, 32, 256)
	6	Zero-Padding	(34, 34, 256)
	7	C512S1-IN-L	(31, 31, 512)
	8	Zero-Padding	(33, 33, 512)
	9	C1S1	(30, 30, 1)

表 8. Pix2Pix 網路架構代號對照

代號名稱	對應意思
CnSm	具有 n 個神經元的卷積層，步進值為 m
L	LeakyReLU 激活函數層
IN	實例正規化層
CTnSm	具有 n 個神經元的反卷積層，步進值為 m
Dn	$n\%$ 的丟棄層
R	ReLU 激活函數層
Concat n	與第 n 層的輸出做合併，用於生成器
Concat n & m	將 n 與 m 層輸出做合併，用於判別器
Tanh	Tanh 激活函數層
Zero-Padding	零填充層

表 9. Pix2Pix 之訓練超參數設定

參數名稱	參數值 (L2+L1)	參數值 (Bce+L1)
訓練批次量	1	2
訓練次數	80000	150000
生成器學習率	0.0002	0.0002
生成器 β_1	0.5	0.5
生成器 β_2	0.999	0.999
判別器學習率	0.0002	0.0002
判別器 β_1	0.5	0.5
判別器 β_2	0.999	0.999
L1損失權重 λ	100	100

$$G_{\text{Pix2Pix}}^* = \arg \min_G \max_D \mathcal{L}_{CGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \quad (8)$$

$$G_{\text{Pix2Pix}} = \mathcal{L}_{L2}(G, D) + \lambda \mathcal{L}_{L1}(G) \quad (9)$$

$$\mathcal{L}_{L2}(G, D) = \frac{1}{f_w f_h} \sum_{i=1}^{f_w} \sum_{j=1}^{f_h} |D(G(x))_{ij} - D(y)_{ij}|^2 \quad (10)$$

$$\mathcal{L}_{L1}(G) = \frac{1}{3wh} \sum_{i=1}^w \sum_{j=1}^h \sum_{k=1}^3 |y_{ijk} - G(x)_{ijk}| \quad (11)$$

3.5 Cycle-GAN

Cycle-GAN 是一個 GAN 的另一種變化，是一種將 GAN 應用於影像風格轉換(Image translation)的非監督式學習演算法。與 Pix2Pix 不同，Cycle-GAN 的輸入資料不是互相成對(pair-to-pair)的資料，即只要提供兩個不同風格的影像資料。因此 Cycle-GAN 演算法可以解決不同風格成對資料獲取困難的問題，本研究使用的模型架構如表 10，架構之層數代號如表 11所示，其中若沒特別說明卷積層的步進值，則步進值為2；LeakyReLU 的負數部分斜率均為0.2；每個卷積層的卷積核大小都為(3,3)。生成器使用深度殘差網路(Res-Net)結構，相較於 U-Net 結構，Res-Net 結構的優點在於參數較少可用直接用殘差區塊連接取代編碼器與解碼器之間的跳接，判別器與 Pix2Pix 一樣使用 PatchGAN 架構。Cycle-GAN 演算法是由兩個 GAN (即兩個生成器與兩個判別器)所組成。其循環生成概念如下圖 8， G_1 、 G_2 為生成器而 D_1 、 D_2 為判別器， G_1 的輸入為真實風格影像，經過模型後會生成草圖風格

的影像； G_2 的輸入則是草圖風格影像，經模型後會生成真實風格影像， D_2 判別 G_1 轉換後的影像 $G_1(x)$ 是否屬於草圖風格； D_1 則判斷經 G_2 轉換後的影像 $G_2(x)$ 是否屬於真實風格。

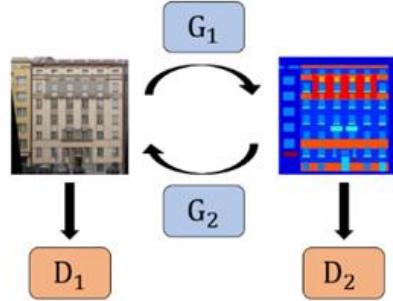


圖 8. Cycle-GAN 循環生成之概念圖

表 10. Cycle-GAN 之網路架構

網路 / 區塊名稱	層數	網路層架構	輸出形狀
殘差網路 R	1	Input	(w, h, c)
	2	Cnk3s1	(w, h, n)
	3	Cnk3s1	(w, h, n)
	4	Add 3 & 1	(w, h, n)
生成器 $G_1 & G_2$	1	Generator-Input	(256, 256, 3)
	2	C64k7s1-IN-ReLU	(256, 256, 64)
	3	C128k3s2-IN-ReLU	(128, 128, 128)
	4	C256k3s2-IN-ReLU	(64, 64, 256)
	5	R256	(64, 64, 256)
	6	R256	(64, 64, 256)
	7	R256	(64, 64, 256)
	8	R256	(64, 64, 256)
	9	R256	(64, 64, 256)
	10	R256	(64, 64, 256)
	11	R256	(64, 64, 256)
	12	R256	(64, 64, 256)
	13	R256	(64, 64, 256)
	14	CT128k3s2-IN-LR	(128, 128, 128)
	15	CT64k3s2-IN-LR	(256, 256, 64)
	16	CT3k7s1-tanh	(256, 256, 3)
判別器 $D_1 & D_2$	1	Discriminator-Input	(256, 256, 3)
	2	C64k4s2-IN-LR	(128, 128, 64)
	3	C128k4s2-IN-LR	(64, 64, 128)
	4	C256k4s2-IN-LR	(32, 32, 256)
	5	C512k4s2-IN-LR	(16, 16, 512)
	6	C1k4s1	(16, 16, 1)

表 11. Cycle-GAN 網路架構代號對照

代號名稱	對應意思
Cnkmsa	神經元為 n 的卷積層，卷積核大小為 (m, m) ，步進值為 a
dk	神經元為 k ，卷積核大小為 $(3, 3)$ 的
Rk	卷積核大小為 $(3, 3)$ ，神經元為 k 的殘差網路
uk	卷積核大小為 $(3, 3)$ ，步進值為 1，神經元為 k 的反卷積層
BN	批次正規化層

LR	LeakyReLU激活函數層
Add n & m	將輸入n與m對應元素相加

Cycle-GAN 生成器的目標函數如(12)所述，其中分成兩個部分，第一個部分是原始 GAN 的目標函數，第二部分則是目標函數展開後的項目(13)，其中又分成對抗損失(Adversarial Loss) (14)、迴圈一致損失(Cycle Consistency Loss) (15)以及特徵損失(Identity Loss) (16)。對抗損失是生成器嘗試生成看起來類似於某一風格的圖像，而判別器則在區分辨識生成器生成之圖像和真實圖片是否為真實圖片並計算誤差，其中公式(14)-(16)中 \mathbb{E} 的意思是指真實樣本分布($y \sim pdata(y)$)與生成樣本($x \sim pdata(x)$)的期望值；迴圈一致損失是計算某個風格影像經過 G_1G_2 如圖 9 所示，對於來自不同風格的每個圖像都應滿足都前後向循環一致性：草圖風格圖像經過生成器 G_2 生成圖像 $G_2(x)$ 再由生成器 G_1 轉回圖像 $G_1(G_2(x))$ 其結果要等同於真實風格的圖像；並計算彼此之間的 L1 距離，可視為原影像與還原影像同位置像素間的差值；而特徵損失則是計算真實圖片經過 G_2 也就是 $G_2(y)$ 、草圖經過 G_1 也就是 $G_1(x)$ 轉換後影像與原始影像的 L1 損失，目的是為了保持轉換後影像的顏色組成，讓 Cycle-GAN 模型只在必要時才對影像作出修改，整體訓練的損失計算如圖 9，本研究的 Cycle-GAN 模型訓練超參數設定如表 12，其中訓練 80000 次，優化器使用 Adam 優化器，所有生成器與判別器學習率皆為 0.0002，其中前 40000 次訓練都為原學習率，後 40000 次訓練每 400 次學習率會遞減 2×10^{-6} ，到訓練結束時學習率剛好會遞減至 0，此舉能夠使訓練後期較穩定。

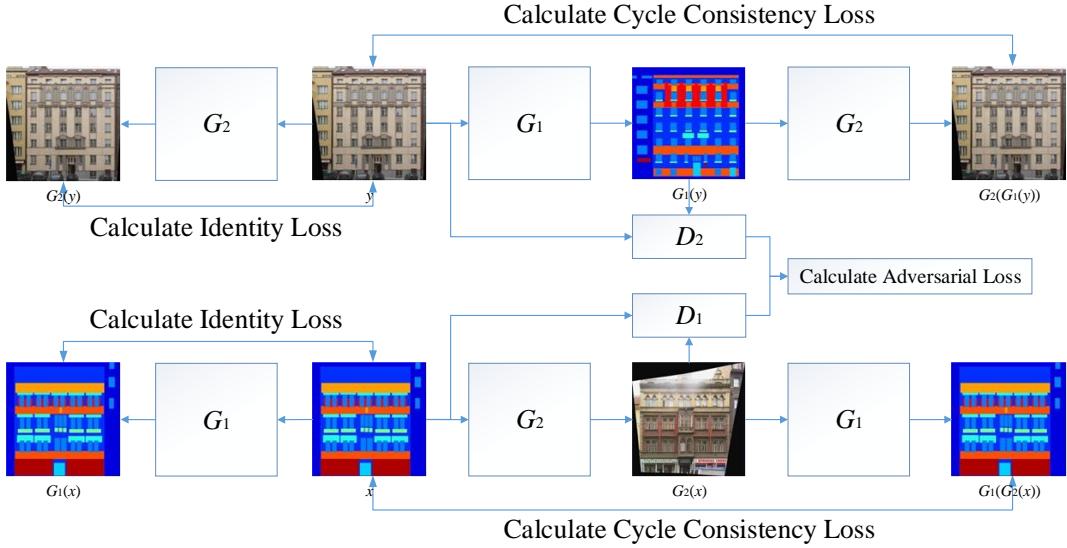


圖 9. Cycle-GAN 訓練與損失計算示意圖

表 12. Cycle-GAN 之訓練超參數設定

參數名稱	參數值
訓練批次量	1
訓練次數	80000
生成器學習率	0.0002
生成器學習率遞減率	2×10^{-6}
生成器 β_1	0.5
生成器 β_2	0.999
判別器學習率	0.0002
判別器學習率遞減率	2×10^{-6}
判別器 β_1	0.5
判別器 β_2	0.999
循環一致性損失權重 λ	10

$$G_1^*, G_2^* = \arg \min_{G_1, G_2} \max_{D_1, D_2} \mathcal{L}(G_1, G_2, D_1, D_2) \quad (12)$$

$$\mathcal{L}(G_1, G_2, D_1, D_2) = \mathcal{L}_{GAN}(G_1, D_2) + \mathcal{L}_{GAN}(G_2, D_1) + \lambda \mathcal{L}_{cyc}(G_1, G_2) + 0.2\lambda \mathcal{L}_{identity}(G_1, G_2) \quad (13)$$

$$\mathcal{L}_{GAN}(G_i, D_j) = \mathbb{E}_{y \sim p_{data}(y)} [(D_j(y) - 1)^2] + \mathbb{E}_{x \sim p_{data}(x)} [D_j(G_i(x))^2] \quad (14)$$

$$\mathcal{L}_{cyc}(G_1, G_2) = \min \mathbb{E}_{x \sim p_{data}(x)} [\|G_2(G_1(x)) - x\|_1] + \min \mathbb{E}_{y \sim p_{data}(y)} [\|G_1(G_2(y)) - y\|_1] \quad (15)$$

$$\mathcal{L}_{identity}(G_1, G_2) = \mathbb{E}_{y \sim p_{data}(y)} [\|G_1(y) - y\|_1] + \mathbb{E}_{x \sim p_{data}(x)} [\|G_2(x) - x\|_1] \quad (16)$$

3.6 去噪擴散隱式模型 (Denoising Diffusion Implicit Models, DDIM)

DDIM [14]是近年來非常引人關注的生成模型，其基礎建立在擴散模型與去噪擴散機率模型上 (Denoising Diffusion Probabilistic Models, DDPM)

[13]。其優秀的生成能力使目前研究都熱衷於探討這個模型，不過此模型的最大缺點就是訓練時間非常久、且使用的運算量相當龐大、也相當看重資料及的數量，若資料及不足則容易因為無法學習到特徵導致訓練失敗。本研究由於資料不足，在訓練條件生成中無法進行良好的訓練，故本研究只針對訓練資料進行隨機生成並探討生成結果，並無條件式生成。

擴散模型的基本原理是利用前向擴散將一張照片添加雜訊，直到圖片變成幾乎成為雜訊；接著訓練神經網路預測此雜訊並使用逆向擴散將雜訊恢復成原始圖片。前向擴散的示意如圖 10，公式如 (17)。其中 $q(x_{1:T}|x_0)$ 為前向擴散中根據原圖 x_0 到擴散時間 $1 \sim T$ 的條件分布形式，也代表原圖與各添加雜訊圖片的對應關係，添加雜訊的雜訊分布為一個常態分布 (19)，其平均為 $\sqrt{1 - \beta_t}$ 、變異數為 β_t ， β 代表擴散時間中的一個數列，通常值由 0 到 1，也是添加雜訊的程度， β_t 即為數列中對應擴散時間 t 的值。

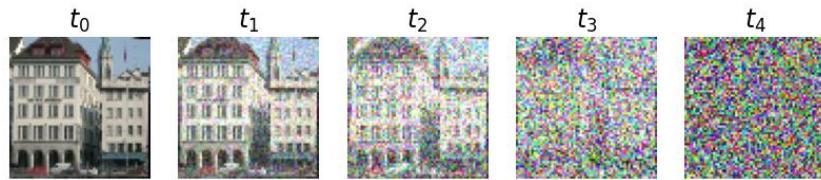


圖 10. 對資料進行前向擴散之示意圖

接著為了學習到由一張雜訊的圖片返回成原圖的辦法，故需要建立一個深度學習模型來學習雜訊分布的狀況，並用於逆向擴散，逆向擴散的分布為 (19)，它也為一個常態分布 (20)，其中將雜訊圖片 x_t 與擴散時間 t 輸入至神經網路計算出 $\mu_\theta(x_t, t)$ 並和 $\sqrt{1 - \beta_t}x_{t-1}$ 進行 KL 散度的相似度計算。至於前向擴散與逆向擴散的變異數經過論文作者實驗，結果非常相似，故不需要計算誤差。擴散模型最終的目標損失函數為 (21)， $\bar{\alpha}_t$ 是用於表達前向擴散到最後的雜訊 x_t 與原圖 x_0 的計算參數； z 為前向擴散之雜訊， z_θ 為模型預測之雜訊。這些公式的詳細推導過程如附錄 4。

DDIM 改良了 DDPM 在前向擴散中使用馬可夫鏈導致採樣時間很長的缺點。做法並非使用馬可夫鏈必須要知道前一個擴散時間的分布才能得知當下擴散時間的分布，如圖 11。它可以透過計算得知以初始條件 t_0 的圖，直接計算出特定時間步的加雜訊圖片。

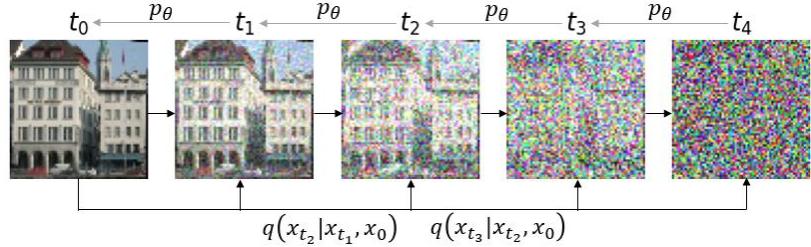


圖 11. DDIM 採樣之示意圖

本研究之 DDIM 模型訓練方式如圖 12 所示，首先定義所需要用到的超參數，接著定義前向擴散與逆向擴散的方法，以及定義模型中使用到的神經網路，研究中使用 U-Net 為主架構。在模型訓練的過程中會先隨機生成一個擴散時間 t 並透過此擴散時間計算對應的雜訊圖片，接著將此雜訊圖片與擴散時間 t 一起被送入神經網路中預測雜訊並計算誤差。在訓練完一次後會評估模型，此時會使用測試資料來測試生成能力並計算生成圖片的 KID 分數，此分數能更好的判斷模型生成能力的變化。最後判斷訓練是否結束，若尚未結束則繼續訓練；結束後生成一批圖片資料用於後續評估，以及儲存所有訓練成果、模型權重等。

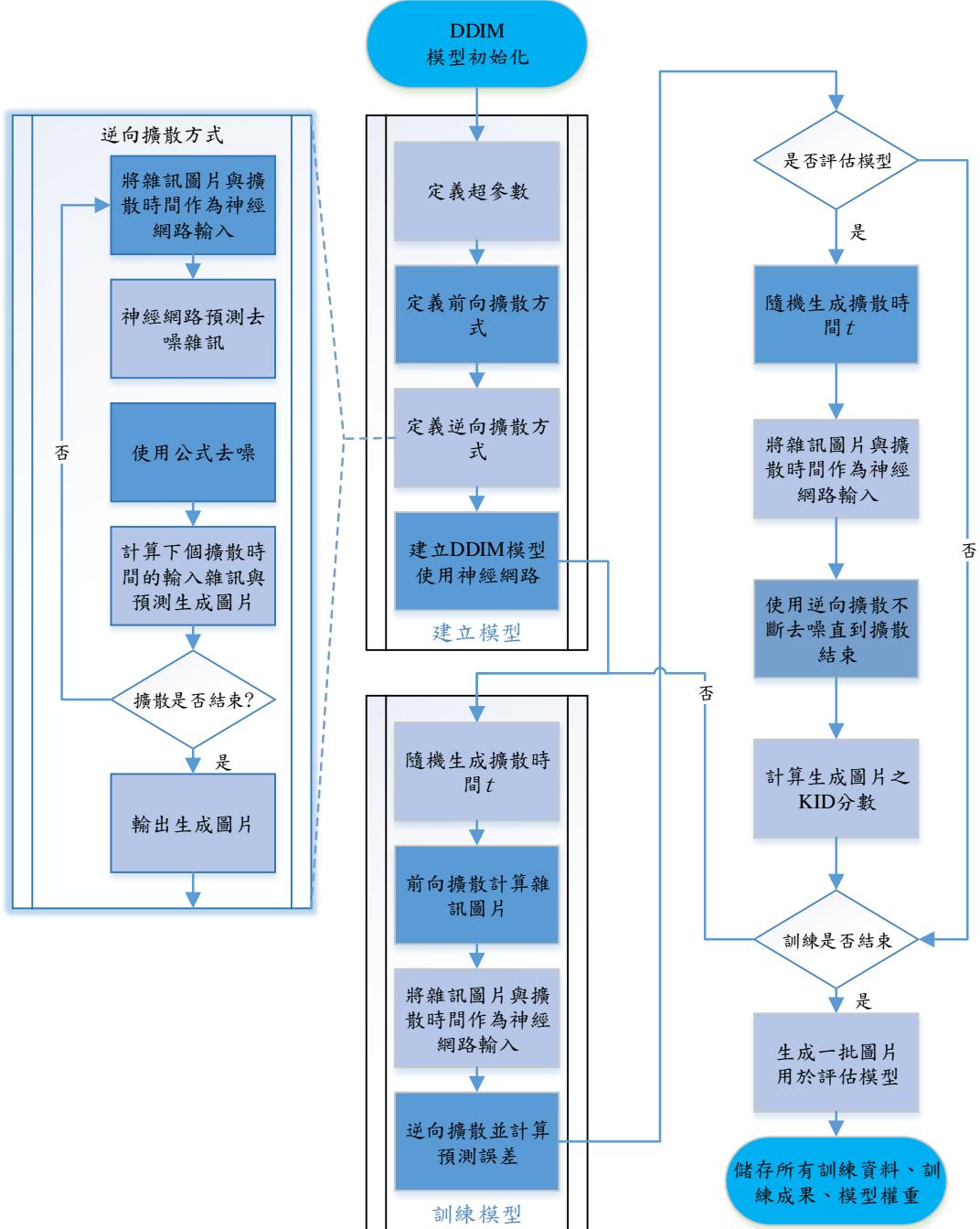


圖 12. DDIM 模型訓練流程

本實驗之 DDIM 模型之擴散時間定義使用餘弦時間表，將擴散時間 t 變成0~1之間的連續值，並非 $t=1,2,3\dots$ 的離散值，首先定義餘弦角度之最大值 $angle_{max}$ 與最小值 $angle_{min}$ ，接著將當下的擴散時間透過 (22) 計算為角度 $angle_t$ 。接著計算這個角度的 sin 值與 cos 值，cos 值對應 $\sqrt{1 - \beta_t}$ ；sin 值對應 $\sqrt{\beta_t}$ 。對應擴散時間 t 下的雜訊圖片則由 (23) 計算，其中 $Image$ 為原始資

料集圖片； \mathcal{N} 為從常態分佈採樣的雜訊。逆向擴散計算方式則使用 (24) 計算， \mathcal{N}_{p_θ} 為神經網路預測之雜訊分布。經過一定次數去噪就能夠將 *Image* 還原成原始圖片。

本實驗 DDIM 之類神經網路架構如下表 13，模型主要使用了 U-Net 架構並搭配殘差區塊建立深層的模型。每個殘差區塊都有兩個卷積層，此外殘差的跳接部分也會再使用一個卷積層轉換。網路架構的代號對照如表 14，網路區塊的輸出形狀中 w, h, c 分別代表輸入資料的寬、高、通道數。上採樣區塊中上採樣層使用雙線性差值法。網路對於擴散時間輸入會經過正弦曲線嵌入層在使用上採樣層把輸出大小擴大成相應尺寸，其中上採樣的插值方法使用最近鄰插值法。另外正弦函數曲線嵌入層會透過將擴散時間 t 作為輸入，接著經過計算嵌入至長度為 32 的向量，計算方法如(25)-(26) 所示。其中(25)代表一個初始值為 0，終止值為 3，公差 d 為 0.2，共有 16 個項目的等差數列。方程式(26)是將這些數列中的所有元素 a_t 都計算出 $2\pi e^{at}$ 的值，再分別計算 sin 與 cos 的值，並整理成變成一個長度為 32 的向量。

本實驗 DDIM 參數使用如下表 15，其中每次訓練的批次量為 2，每個 epoch 會完整的訓練一次資料集，故只有 50 輪的訓練。優化器使用 AdamW，這是 Adam 的變種之一，其加入了權重衰減係數，值為 0.00005。其他參數與 Adam 相同，學習率為 0.0001； β_1 為 0.5； β_2 為 0.999。另外擴散時間使用餘弦擴散時間表，故會限制其角度的上下限，角度最大值為 $\cos^{-1}(0.95)$ ，也就是約 0.31756 rad，角度最小值為 $\cos^{-1}(0.02)$ ，也就是約 1.55079 rad。

表 13. DDIM 之網路架構

網路 / 區塊名稱	層數	網路層架構	輸出形狀
殘差區塊 R	1	Input	(w, h, c)
	2-1-1	BN	(w, h, c)
	2-1-2	Cnk3	(w, h, n)
	2-1-3	Swish	(w, h, n)
	2-1-4	Cnk3	(w, h, n)
	2-2	Cnk1	(w, h, n)
下採樣區塊	3	Add 2-1-4 & 2-2	(w, h, n)
	1	Input	(w, h, c)

D	2	Rn	(w, h, n)
	3	Rn (Concat input)	(w, h, n)
	4	AveragePooling Size=2	(w/2, h/2, n)
	1	Input	(w, h, c)
上採樣區塊 U	2	Up-Sampling Size=2 Interpolation used “bilinear”	(2w, 2h, c)
	3	Concat D ₃	(2w, 2h, c+c _{D3})
	4	Rn	(2w, 2h, n)
	5	Rn	(2w, 2h, n)
	1-1	Noise input	(64, 64, 3)
DDIM 網路架構	1-1-2	C32k1	(64, 64, 32)
	1-2	Diffusion time input	(1, 1, 1)
	1-2-1	Sinusoidal embedding	(1, 1, 32)
	1-2-2	Up-Sampling Interpolation used “nearest”	(64, 64, 32)
	2	Concat 1-1-3 & 1-2-2	(64, 64, 64)
	3	D64	(32, 32, 64)
	4	D128	(16, 16, 128)
	5	D256	(8, 8, 256)
	6	D256	(4, 4, 256)
	7	R512	(4, 4, 512)
	8	R512	(4, 4, 512)
	9	U256 Concat 6	(8, 8, 256)
	10	U256 Concat 5	(16, 16, 256)
	11	U128 Concat 4	(32, 32, 128)
	12	U64 Concat 3	(64, 64, 3)
	13	C3k1 Output	(64, 64, 3)

表 14. DDIM 網路架構代號對照

代號名稱	對應意思
Cnkm	具有 n 個神經元的卷積層，卷積核大小為 (m, m)
Swich	swish激活函數層，函數公式如(27)
BN	批次正規化層
Add n & m	將輸入 n 與 m 對應元素相加
Concat n & m	將 n 與 m 層輸出做合併，用於判別器
Sinusoidal embedding	正弦曲線嵌入層
Up-Sampling	上採樣層，將輸入寬高放大size倍
AveragePooling	平均池化層，將輸入寬高縮小size倍
Concat c _{D3}	上採樣與對應下採樣第3層的輸出做合併
Dn	下採樣區塊，其中每個卷積層都有 n 個神經元
Rn	殘差區塊，其中每個卷積層都有 n 個神經元
Un	上採樣區塊，其中每個卷積層都有 n 個神經元

表 15. DDIM 之訓練超參數設定

參數名稱	參數值
訓練批次量	2
訓練次數	50
學習率	0.0001
優化器 β_1	0.5
優化器 β_2	0.999
權重衰減係數	0.00005

擴散時間之 $angle_{min}$	$\cos^{-1}(0.02)$
擴散時間之 $angle_{max}$	$\cos^{-1}(0.95)$

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^T q(x_t|x_{t-1}) \quad (17)$$

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \quad (18)$$

$$p(x_{0:T}) = p(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t) \quad (19)$$

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \quad (20)$$

$$L_{simple}(\theta) = \mathbb{E}_{t,x_0,z} \left[\|z - z_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z, t)\|^2 \right] \quad (21)$$

$$angle_t = angle_{min} + t(angle_{max} - angle_{min}) \quad (22)$$

$$\mathbf{N}_t = \sqrt{1-\beta_t} \cdot \text{Image} + \sqrt{\beta_t} \cdot \mathcal{N} \quad (23)$$

$$\text{Image} = \frac{N_t - \sqrt{\beta_t} \cdot \mathcal{N}_{p_\theta}}{\sqrt{1-\beta_t}} \quad (24)$$

$$a_t = \overbrace{\{0, 0.2, 0.4, \dots, 3\}}^{d=0.2, n=16} \quad (25)$$

$$e_t = \overbrace{\{\sin(2\pi e^{a_t}), \cos(2\pi e^{a_t})\}}^{n=32} \quad (26)$$

$$swish(x) = \frac{x}{1 + e^{-x}} \quad (27)$$

3.7 Adam 優化器

Adam 優化器[17]是深度學習中常用的優化器之一，也是本研究中使用的優化器。Adam 優化器結合了其他優化器例如 RMSProp 與 AdaGrad 的優點，其他優點包括可以自動調整學習率、可應用於不穩定的目標函數、相比其他優化器較不容易陷入局部最佳等。此優化器會對梯度的平均值與梯度的變異數進行考慮並計算下次更新的值。

Adam 優化器更新權重參數的方式如方程式(28)-(36)。方程式 (28)是第 t 訓練時間的梯度，其中 θ 是訓練的深度學習模型內的參數。根據這個梯度接著計算梯度的一階矩估計值 (First moment estimate) (29)，初始化的 m_0 為 0，其中 β_1 是可設定的超參數，用於控制權重的分配。若 β_1 值接近0則會較依賴當前的梯度，反之接近1則會較依賴於先前的權重。接著使用方程式

(30) 計算梯度的二階矩估計值 (Second raw moment estimate)，初始化的 v_0 也等於 0，公式中的 β_2 可以控制梯度平方的影響狀況，接近 1 會更依賴於之前訓練之梯度平方估計，會使訓練初期時能夠比較穩定。由於 m_0 和 v_0 為 0 所以會導致 m_t 與 v_t 偏向於 0，所以需要對 m_t 與 v_t 進行偏差的修正，降低這些偏差對於訓練的影響，修正的方式分別為 (31) 和 (32)。最後則是更新權重參數的步驟，這一步會使用 (33) 來進行更新，其中 α_{lr} 為學習率； ϵ 為一個很小的值，為了避免 \hat{v}_t 為 0 造成分母為 0 而使更新變的無意義，通常預設值為 10^{-8} 。

$$g_t = \nabla_{\theta} f(\theta_{t-1}) \quad (28)$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (29)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (30)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad (31)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (32)$$

$$\theta_t = \theta_{t-1} - \frac{\alpha_{lr} \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \quad (33)$$

3.8 模型評估指標

本研究使用多種不同的模型評估指標來評估生成模型生成圖片之圖片品質。這些指標各自都有特性，也被應用於不同任務中。本研究同時使用這些指標來判斷生成模型對於訓練資料與測試資料的生成能力，並藉此來比較不同模型在風格變換任務中的訓練成果、生成能力。值計上來說會針對模型生成圖片的像素差異、結構差異以及圖片特徵的差異進行全面性的比較。

3.8.1 結構相似性指數

結構相似性指數 (Structural Similarity Index, SSIM)，為比較基礎的相似度指標，其顧名思義是用於計算兩張圖片之間其結構的相似性，此指標以圖片的亮度、對比度以及結構為計算的核心，透過考慮這三個因素來計算圖片失真的程度，這個指標因為考慮結構等因素，所以計算出來的結果會更符合人類的感知。

SSIM 的計算分為三個部分，分別為：亮度 (34)、對比度 (35)、結構 (36)。以這三個部分的結果為基礎再來計算 SSIM (37)。亮度計算中， μ_x 與 μ_y 分別為圖片 x 與圖片 y 的像素平均值， C_1 為常數；對比度計算中， σ_x 與 σ_y 為圖片的標準差， C_2 為常數；結構計算中， σ_{xy} 為兩張圖片的共變異數， C_3 為常數。最後 SSIM 計算中 α 、 β 、 γ 都是超參數，可以決定指標中各部分的權重值。

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \quad (34)$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \quad (35)$$

$$s(x, y) = \frac{2\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \quad (36)$$

$$SSIM(x, y) = [l(x, y)]^\alpha [c(x, y)]^\beta [s(x, y)]^\gamma \quad (37)$$

3.8.2 峰值訊噪比

峰值訊噪比 (Peak Signal-to-Noise Ratio, PSNR) 是用來評估兩張圖片相似程度的指標。這個指標將圖片以訊號處理的方式計算其相似度，將圖片訊號以 PSNR 計算後的單位可視為分貝數，並依據分貝大小來判斷圖片的相似度。通常結果為 30dB~50dB 時圖片差異肉眼較難看出；低於 30dB 時肉眼可以明顯看得出圖片的不同。

PSNR 的其公式為 (39)。其中 x 為生成圖片、 y 為真實圖片。 Max_y^2 是真實圖片中像素最大值的平方，乘以 3 是因為圖片為彩色，色彩通道為 3。計算完成後會除以真實圖片與生成圖片的 L2 誤差，也就是 $MSE(x, y)$ ，其計算方式如 (38)。其中 w 為圖片的寬度， h 為圖片的高度，接著將兩張圖片中每個像素的差異平方計算出來再計算算數平均，即為所求。

$$MSE(x, y) = \frac{1}{3wh} \sum_{i=1}^w \sum_{j=1}^h \sum_{k=1}^3 (x_{i,j,k} - y_{i,j,k})^2 \quad (38)$$

$$PSNR(x, y) = 10 \log_{10} \left(\frac{3Max_y^2}{MSE(x, y)} \right) \quad (39)$$

3.8.3 Kernel Inception Distance (KID)

KID 是基於 Inception 網路的計算方式，Inception 網路是一個已經訓練好的深度學習模型，其使用 ImageNet 資料集[18]來做學習，資料集總計有 1000 種物件的分類，資料量共超過一百萬張圖片。但在計算 KID 時只會使用其中部分網路層以得到圖片的特徵圖。

KID 可以透過計算 Inception 網路出來的圖片特徵，將生成圖片與真實圖片特徵的平均值差異之平方計算出來並衡量兩個特徵之間的差異。此外 KID 還有一個三次核的無偏估計值，這個估計值能夠讓計算出來的結果更貼近人類的感知。所以 KID 總結來說即為將 Inception 網路計算出來之特徵向量空間的多項式核函數平方的最大平均差異 (Maximum Mean Discrepancy, MMD) (40)。其中 P 與 Q 分別代表生成圖像分布與真實圖像分布； H 為一個希爾伯特空間。方程式的目的為希望找到一個函數 f ，其中 f 期望兩個機率分布 $E_p f(x)$ 與 $E_Q f(y)$ 的動差相減可以達到最大值，即求得期望之最小上界 (sup)。其中 f 屬於希爾伯特空間 H ，且 f 限制在再生希爾伯特空間，其單位球 $|f|_H \leq 1$ 。接著使用里斯表示定理 (Riesz's Representation theorem) 將期望 $E_p f(x)$ 改寫成希爾伯特空間的內積形式，如 (41)。其中 μ_p 為平均嵌入 (Mean Embedding)[19]，此時 MMD 可視為計算兩個分布在某個空間中 Mean Embedding 的距離。為了使這個距離最大，可以利用在希爾伯特空間中範數就是內積的原理來將原本的內積計算變成範數。然後在再生希爾伯特空間中內積可以使用核 (kernel) 計算，最後整個公式可以變成平方 MMD 的表達式 (45)，也就是 KID 的計算方式。其中 m 為生成圖片數量； n 為真實圖片數量； $k(a, b)$ 為核 (kernel) 的公式 (42)； d 為特徵向量維度，本實驗中使用 2048。

$$MMD(P, Q, H) = \sup_{f \in H, |f|_H \leq 1} E_p f(x) - E_Q f(y) \quad (40)$$

$$E_p f(x) \leq f, H < E(\emptyset(x)) \leq f, \mu_p > H \quad (41)$$

$$k(a, b) = \left(\frac{1}{d} a^T b + 1 \right)^3 \quad (42)$$

$$f_m(x) = \frac{1}{m(m-1)} \sum_{i \neq j}^m k(x_i, x_j) \quad (43)$$

$$f_n(y) = \frac{1}{n(n-1)} \sum_{i \neq j}^n k(y_i, y_j) \quad (44)$$

$$MMD^2(x, y) = f_m(x) + f_n(y) + \frac{2}{mn} \sum_{i=1}^m \sum_{j=1}^n k(x_i, y_j) \quad (45)$$

3.8.4 Fréchet Inception Distance (FID)

FID 分數是非常廣泛的圖像生成模型判斷的指標之一，其基礎也使用了 Inception 網路來萃取圖片的特徵。FID 適合評分生成模型的多樣性，但因為模型基於特徵提取所以並不會考慮特徵位置的合理性。比起 KID，FID 計算量較少，所以還是比較廣泛用於生成模型的訓練，FID 分數越低代表模型生成圖片之質量較佳。

FID 分數計算之公式為 (46)，其中 \hat{x} 與 \hat{y} 是生成圖片與真實圖片經過 Inception 網路計算出來的圖片特徵； μ 為平均值，公式會計算平均值差距的平方；接著會計算特徵的共變異數 σ ，經過計算過後出來的結果為方陣，需要再計算其跡數 (Trace)。最後將總結果相加即為 FID 分數。

$$FID(x, y) = |\mu_{\hat{x}} - \mu_{\hat{y}}|^2 + Tr(\sigma_{\hat{x}} + \sigma_{\hat{y}} - 2\sqrt{\sigma_{\hat{x}}\sigma_{\hat{y}}}) \quad (46)$$

3.8.5 Learned Perceptual Image Patch Similarity (LPIPS)

Learned perceptual image patch similarity (LPIPS)[20]為計算兩張圖片的感知損失，這個指標也是基於深度學習模型提取特徵後的特徵相似度。比起不使用深度學習的 PSNR 與 SSIM，LPIPS 也能更貼近人類的感知。LPIPS 計算出來的值越低代表圖片的相似程度越高。

根據其原始論文，LPIPS 的公式為 (47)。其中 1 為從神經網路的第一層提取圖片特徵，本實驗的神經網路使用 11 層的 AlexNet，模型架構如表 16，本研究中所使用之 LPIPS 函式庫為使用 Pytorch 開發的，在卷積運算上與 Keras 略有不同。Pytorch 中卷積運算出特徵的長寬如 (48)，其中 S_{Output} 為輸出特徵長寬； S_{Input} 為輸入資料長寬； p 為填充大小； k 為卷積核大小； $stride$ 為步進值。最大池化層的計算一樣如 (48) 所示。AlexNet 於 2012 年被發表。它當時在 ImageNet 資料集上展現了最先進的性能，用於 LPIPS 時速度最快且效能最好。 H_l 跟 W_l 分別是該層出來的圖片特徵高與寬，皆為 15，作者使用此方法在通道維度上進行單位正規化。公式中使用向量 w_l 對每個

通道中經過激活函數計算的結果進行縮放，並計算 L2距離。研究中通道數 w_l 為 256。

$$LPIPS(x, y) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \left\| w_l \odot (\hat{x}_{h,w}^l - \hat{y}_{h,w}^l) \right\|_2^2 \quad (47)$$

$$S_{Output} = \frac{S_{Input} + 2 * p - (k - 1) - 1}{stride} + 1 \quad (48)$$

表 16. AlexNet 之網路架構

層數	網路層架構	輸出形狀
1	Image input	(3, 256, 256)
2	C64K11S4P2	(64, 63, 63)
3	ReLU	(64, 63, 63)
4	Mk3s2p1	(64, 31, 31)
5	C192k5s1p2	(192, 31, 31)
6	ReLU	(192, 31, 31)
7	Mk3s2	(192, 15, 15)
8	C384k3s1p1	(384, 15, 15)
9	ReLU	(384, 15, 15)
10	C256k3s1p1	(256, 15, 15)
11	ReLU	(256, 15, 15)
12	C256k3s1p1 Output	(256, 15, 15)

表 17. AlexNet 網路架構代號對照

代號名稱	對應意思
CnKmSaPb	具有 n 個神經元的卷積層，卷積核大小為 (m,m) ，步進值為 a ，填充大小為 b
ReLU	ReLU 激活函數層
MKnSmPa	核大小為 (n,n) ，步進值為 m ，填充大小為 a 的最大池化層

第4章 研究成果

4.1 實驗結果

本研究在經過大量實驗與修改後對所有的生成模型進行了完整的實驗並記錄結果。在生成圖片的過程中，每種模型都因其自身的特性而導致結果有所不同。總得來說 Pix2Pix 在面對看過的圖片其生成能力最好，但面對測試資料生成效果就會有瑕疵。Cycle-GAN 在訓練過後可以學習到資料集的特徵分布與組成，在面對測試資料及則可以生成訓練中學習到的風格。而 CGAN 效能最差，無論是訓練資料與測試資料其生成效果都不佳，且在訓練過程中常常會伴隨著梯度消失或者梯度爆炸的問題。

4.1.1 CVAE 生成結果探討

本研究採用 CVAE 進行圖像翻譯生成任務，將草圖資料輸入模型中以生成真實圖片，並使用 PSNR、SSIM 等指標評估生成結果訓練結果之評估分數如表 18。訓練結果顯示，PSNR 為14.387、SSIM 為0.3、FID 為6.64、KID 為0.227，LPIPS 為0.355。值得注意的是，PSNR 和 SSIM 值相對較低，可能表示生成圖像在像素與結構上與真實圖像存在較大的差異，但 FID、KID 分數較低所以在圖片特徵的判斷上比較接近真實圖片。

表 18. CVAE 訓練結果之評估分數

CVAE	
PSNR	14.387
SSIM	0.3
FID	6.64
KID	0.227
LPIPS	0.355

CVAE 的訓練過程如圖 13所示，分別展示了第50、200、1000、5000、9000次訓練的生成效果。在前100次訓練中，CVAE 尚未完全捕捉圖片細節，因此生成的圖片較模糊。到第200次訓練時，模型已學習到圖片特徵，可以清晰地呈現建築物外觀。經過1000次訓練後，生成的圖片逐漸清晰，能夠捕捉每張圖片的特徵，然而在較細緻的內容如樹枝方面，CVAE 仍難以清晰捕捉，呈現較模糊的結果。未來的改進方向可以考慮使用 CVAE-

GAN 以生成更細緻的圖片，使用對抗性的訓練可以更好的讓生成器學習到真實資料分布的情形，也能進一步生生成圖片的品質。

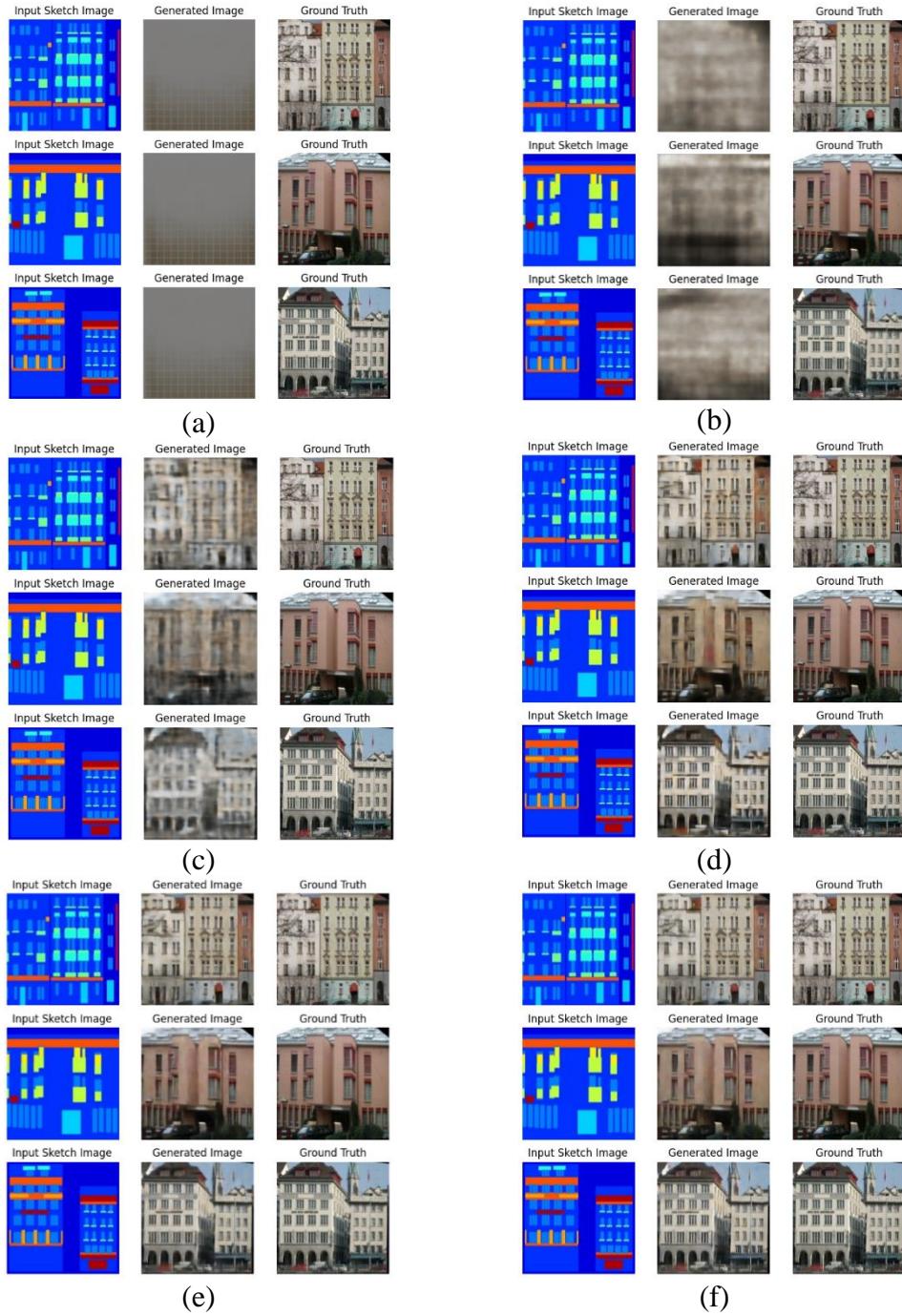


圖 13. 本研究建立之 CVAE 對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第50次訓練, (c) 第200次訓練, (d) 第1000次訓練, (e) 第5000次訓練, (f) 第9000次訓練。

訓練損失如圖 14 所示，初期損失極高，表示 CVAE 模型難以生成相應的圖片。隨著訓練進行，損失逐漸下降，在訓練約 6000 次後趨於穩定，顯示模型能夠生成與真實圖片相似的結果。圖 15 展示了測試資料的生成結果。在訓練過程中生成的圖片逐漸逼近真實圖片，然而在使用測試資料集時呈現相對模糊的效果。總體而言，CVAE 對於圖像翻譯生成任務具有可行性，但受限於資料數量不足可能導致過度擬合的問題。

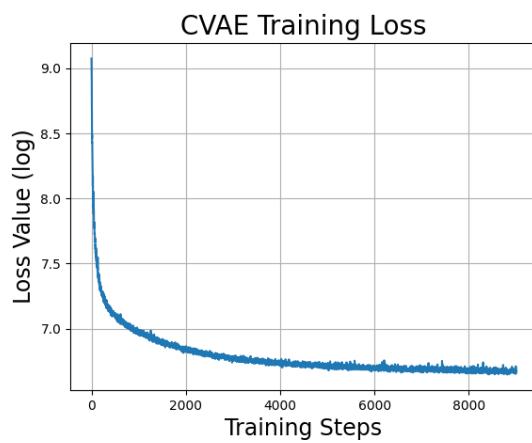


圖 14. CVAE 訓練之損失變化

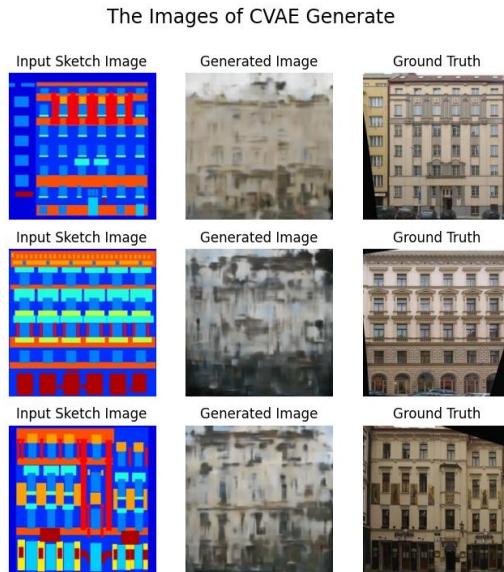


圖 15. CVAE 對於測試資料集的生成能力結果

4.1.2 CGAN 生成結果探討

本研究之 CGAN 生成訓練使用一種損失進行訓練，使用了與原始論文相同的設定，即 CGAN 的判別器損失函數使用二元交叉熵損失 (BCE

Loss)；生成器的對抗損失使用二元交叉熵損失。CGAN 訓練如圖 15所示，在訓練中使用了30000次的訓練時長。由圖可知在大約1000次訓練時有學習到些微的輪廓，在5000次訓練時訓練就差不多收斂，且出現了梯度消失的情況，以至於接著直到30000次訓練結束時生成的圖片結果都並無太大的差異，CGAN 總訓練時長為16979.73秒。CGAN 的生成能力經過不同指標評估的結果如表 19，可知 PSNR 為7.492；SSIM 為0.124；FID 為16.111；KID 為0.433；LPIPS 為0.696。這意味著 CGAN 在圖片像素的相似度、圖片的結構相似度、以及圖片特徵的相似度上都與真實圖片有著較大的差異，訓練結果整體而言比較差。

表 19. CGAN 訓練結果之評估分數

CGAN	
PSNR	7.492
SSIM	0.124
FID	16.111
KID	0.443
LPIPS	0.696

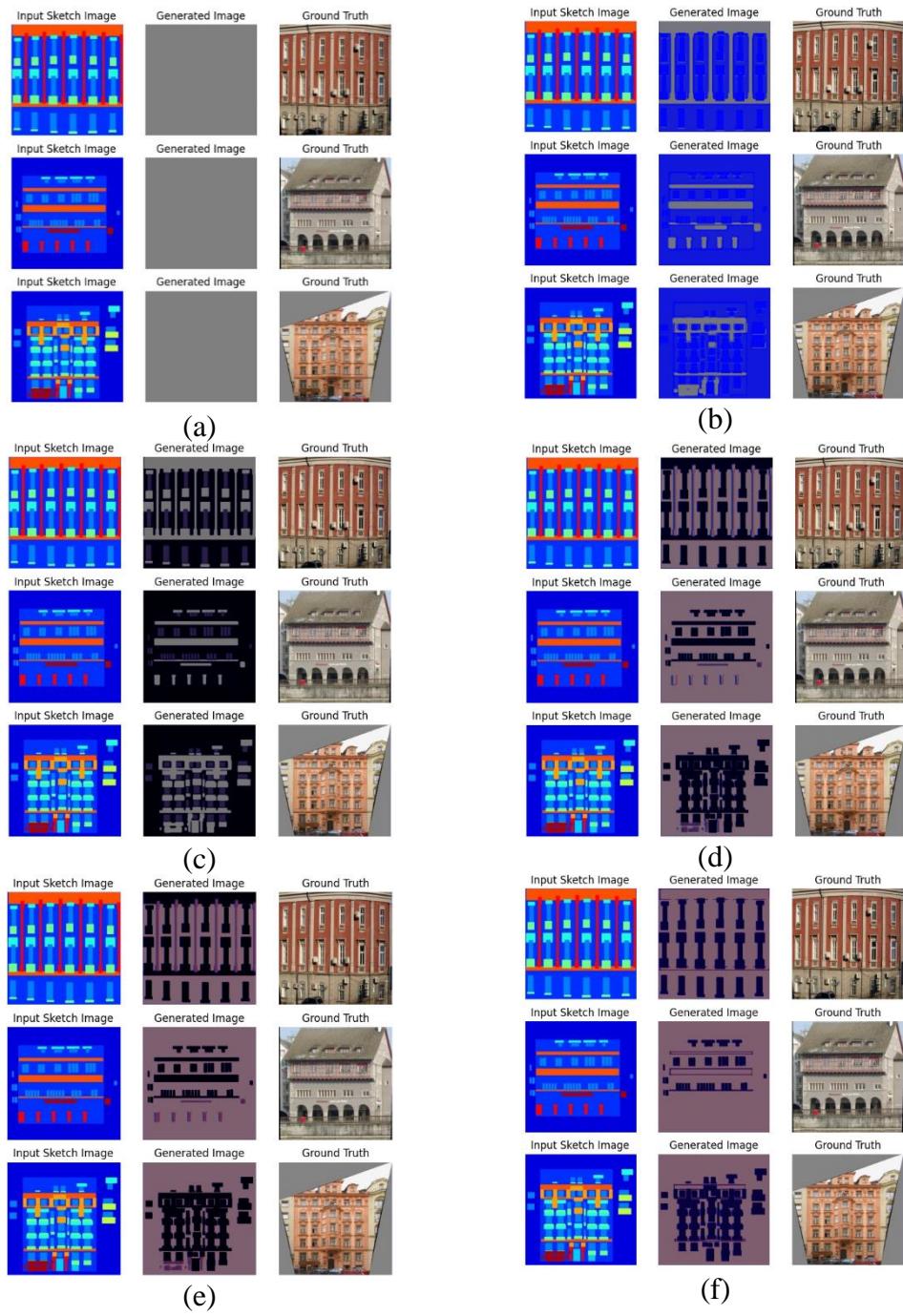


圖 16. 本研究建立之 CGAN 對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第200次訓練, (c) 第1000次訓練, (d) 第5000次訓練, (e) 第10000次訓練, (f) 第30000次訓練。

本研究中訓練損失如圖 17，為 CGAN 在對抗式訓練下的損失變化，由圖可知由於生成器與判別器的對抗式訓練，導致訓練損失較不穩定，另外訓練因為中前期梯度消失所以後續訓練對於優化的效果就沒有太大的助益。生成器與判別器在訓練約6000次時有出現梯度爆炸的情形，所以在後

續對抗訓練中已經失衡，無論如何訓練都無法收斂到最佳的收斂點。在未來或許可以使用多階段的訓練一步一步訓練生成器生成較高品質的結果。

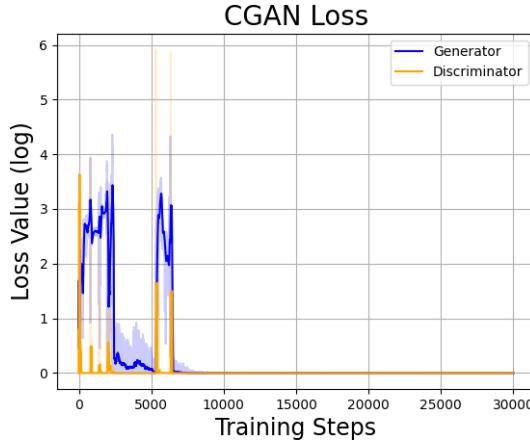


圖 17. CGAN 訓練之對抗損失

本研究對於測試資料之生成能力所生成的預測圖如圖 18，生成的圖片因為訓練效果不好導致最終圖像較不可辨識，只有輪廓勉強可以辨識，大致圖像有點類似草圖，整體而言 CGAN 的訓練效果都比較差。

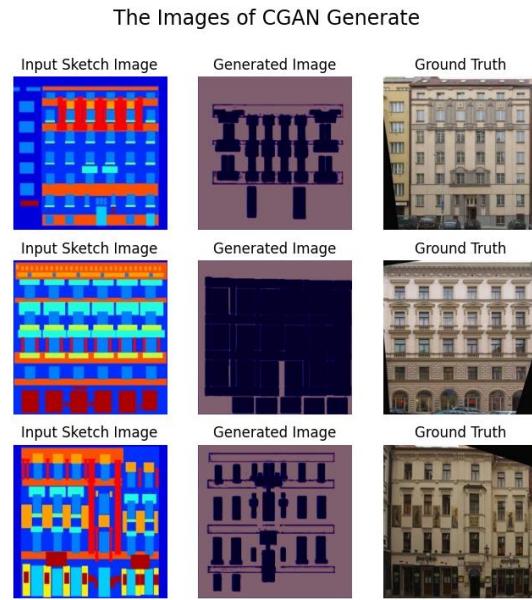


圖 18. CGAN 對於測試資料之生成能力預測圖

4.1.3 Pix2Pix 生成結果探討

本研究之 Pix2Pix 生成訓練使用兩種不同的損失進行訓練，並得到了不同的結果。第一種是與原始論文相同的設定，即 Pix2Pix 的判別器損失函數使用二元交叉熵損失 (BCE Loss)；生成器的對抗損失使用二元交叉熵損失，對於圖片生成的判斷使用 L1 損失。第二種是多方嘗試後修改的設定，

即 Pix2Pix 的判別器損失函數使用 L2 損失；生成器的對抗損失使用 L2，對於圖片的判斷一樣使用 L1 損失。原始 Pix2Pix 訓練如圖 19 所示，在原始的訓練中使用了 150000 次的訓練時長。由圖可知在大約 100000 次訓練時邊界仍有一點模糊，直到 150000 次訓練左右生成器才有學習到轉換後圖片的特徵，訓練時間為 32758 秒。研究中所修改的方法對於訓練資料的生成如圖 20 所示，可以看到在訓練中約 20000 次時生成器就幾乎學會生成訓練資料的分布了，但也因為這個高度擬合性所以在對於測試資料這些沒訓練過的圖片生成則會接近於訓練資料集，也就是說生成器可能過度擬合而造成無法生成具有多樣性的結果。表 20 為兩種不同目標函數訓練之 Pix2Pix 的生成能力評估，由表可知在所有指標的計算下，本研究改寫的目標函數訓練出來的模型生成能力都優於原始的目標函數訓練出來的模型。

表 20. Pix2Pix 訓練結果之評估分數

	Pix2Pix (L2+L1)	Pix2Pix (Bce+L1)
PSNR	23.725	13.675
SSIM	0.588	0.228
FID	4.453	14.063
KID	0.027	0.329
LPIPS	0.186	0.431

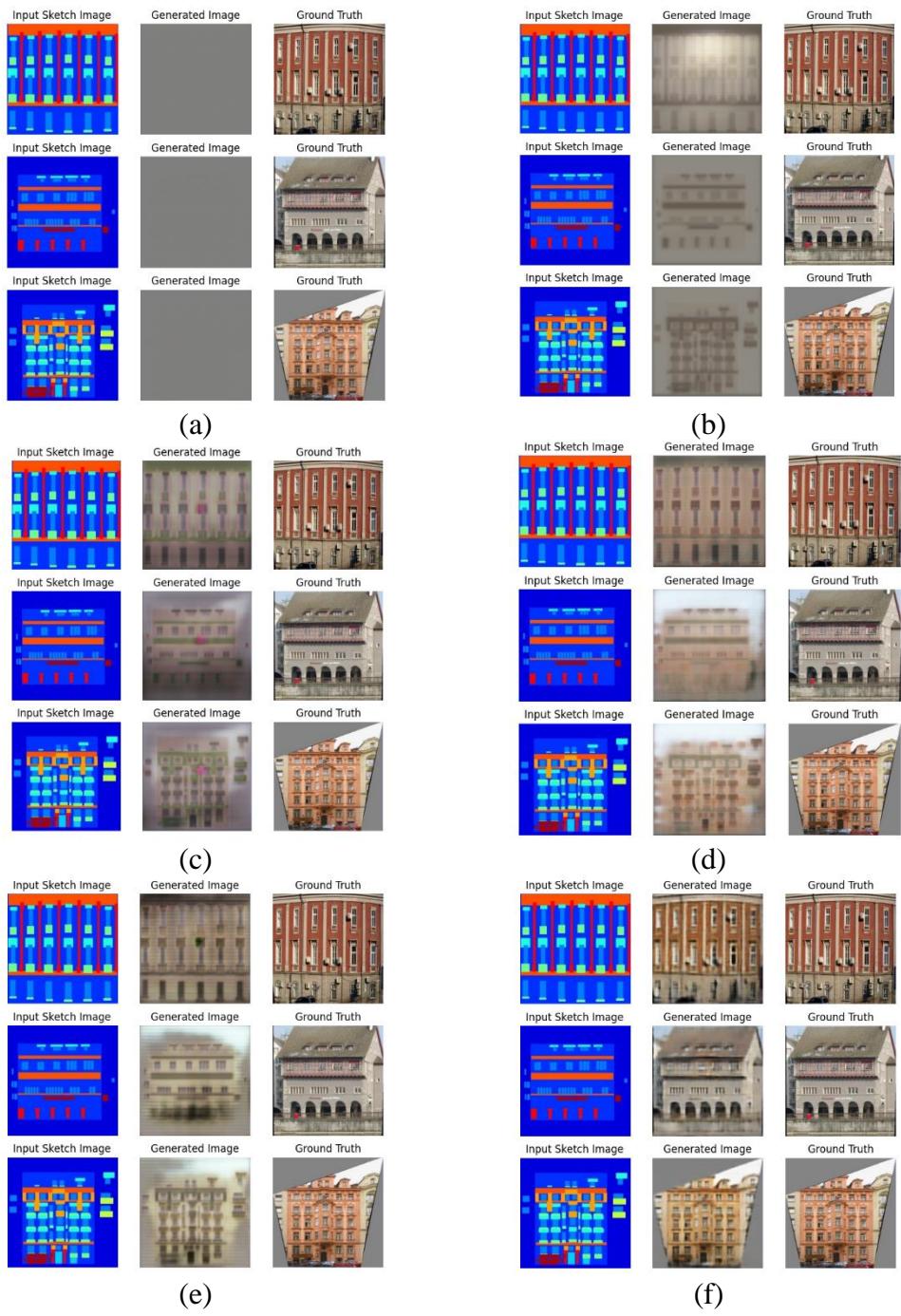


圖 19. 原始論文之 Pix2Pix 對於訓練資料之訓練情形 (Bce+L1), (a) 第 1 次訓練, (b) 第 1000 次訓練, (c) 第 10000 次訓練, (d) 第 25000 次訓練, (e) 第 50000 次訓練, (f) 第 150000 次訓練。

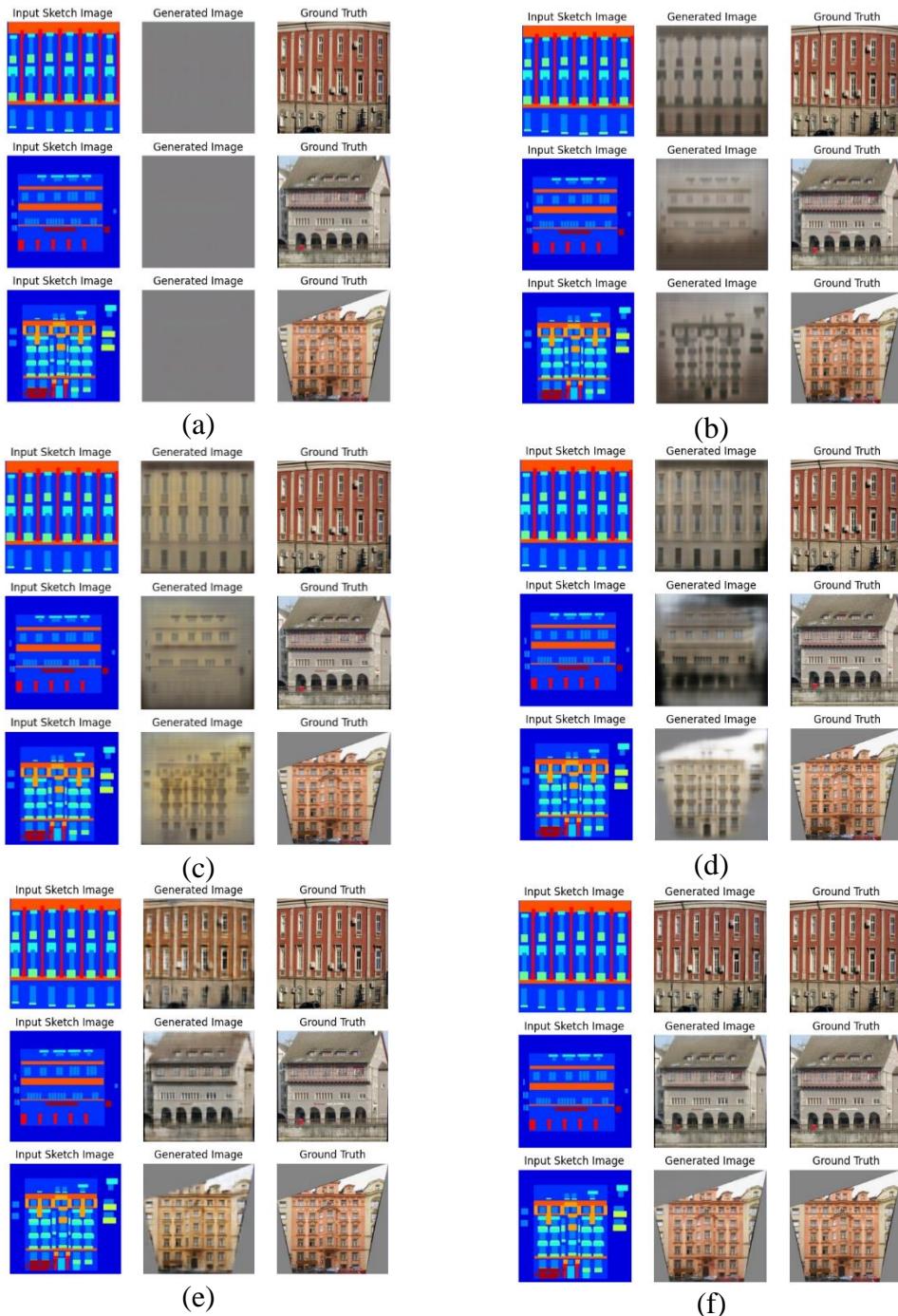


圖 20. 本研究修改之 Pix2Pix 對於訓練資料之訓練情形 (L2+L1), (a) 第 1 次訓練, (b) 第 200 次訓練, (c) 第 1000 次訓練, (d) 第 2500 次訓練, (e) 第 10000 次訓練, (f) 第 20000 次訓練。

本研究中訓練損失如圖 21與圖 22，圖 21為原始論文設定之目標函數訓練下的損失函數變化，由圖可知由於生成器與判別器的對抗式訓練，導致訓練損失較不穩定。另外生成器的像素損失也因為其對抗式的訓練導致較不穩定，不過整體是有下降的趨勢，代表生成器生成的圖片越來越逼近

真實圖片。圖 22是本研究修改目標函數過後的訓練損失變化，由圖可知，生成器與判別器在訓練約10000次時判別器就訓練的比生成器強，所以在對抗訓練中已經失衡。但是生成器的像素損失仍會讓生成器繼續生成逼近真實圖片的影像，因為對抗訓練失衡，所以在像素損失訓練上有較平穩下降的趨勢，之後生成的圖片很逼近真實圖片。不過這樣訓練在資料不充足的情況下會造成過度擬合。綜觀來說原始論文設定時訓練出來的圖片對於訓練資料集的生成有時會有肉眼可見的瑕疵；對於測試資料集的生成則較模糊。但使用修改過後的設定則對於訓練訓練集的擬合狀態較佳；對於測試資料集的生成則有一些瑕疵。對於測試資料的生成結果如圖 23。

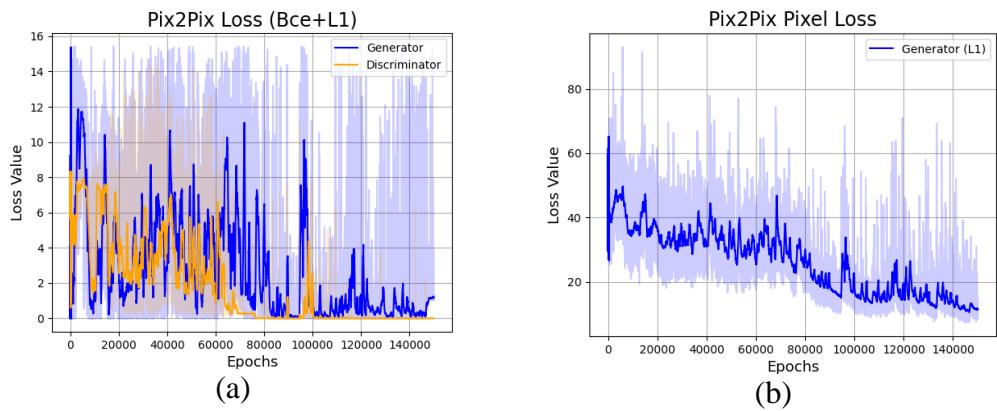


圖 21. 原始論文之 Pix2Pix 訓練損失, (a) 對抗損失, (b) 像素損失

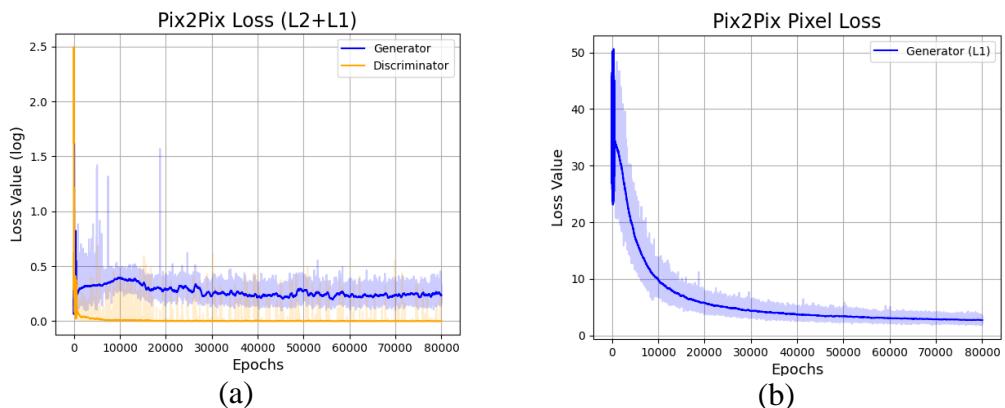


圖 22. 本研究修改之 Pix2Pix 訓練損失, (a) 對抗損失, (b) 像素損失

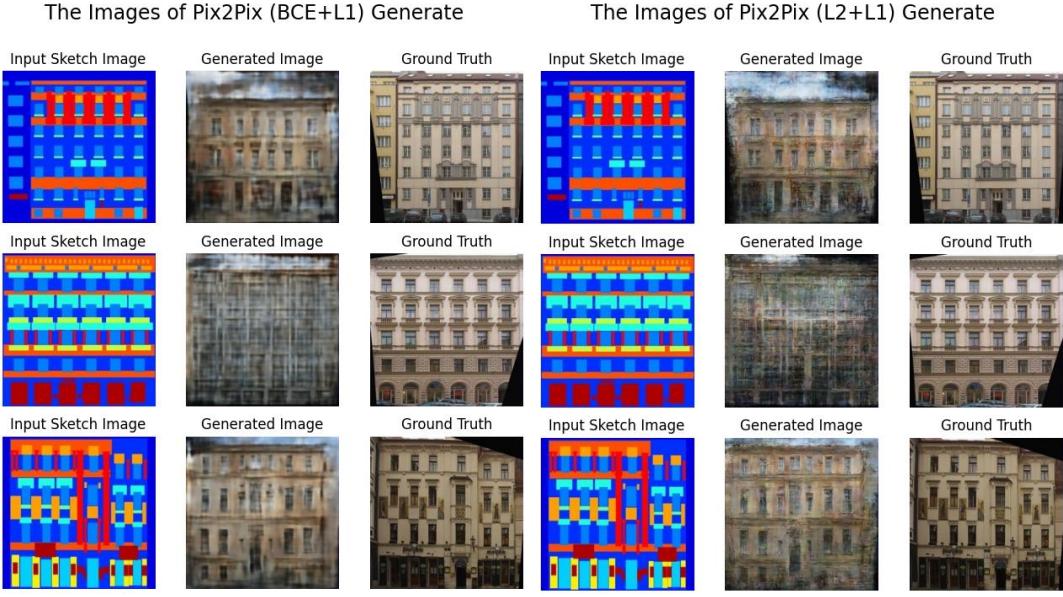


圖 23. Pix2Pix 對於測試資料之生成能力比較, (a) 生成器損失使用 BCE+L1, (b) 生成器損失使用 L2+L1。

4.1.4 Cycle-GAN 生成結果探討

本研究之 Cycle-GAN，為單純使用殘差網路(ResNet)架構，使用此架構訓練能夠避免梯度消失，Cycle-GAN 訓練結束後生成的圖片經過不同指標評估的結果如表 21 所示，其中 PSNR 為 9.812，SSIM 為 0.055，FID 為 8.999，KID 為 0.14，LPIPS 為 0.375。雖然 Cycle-GAN 並不會生成與真實圖片完全一致的圖片，導致 PSNR 與 SSIM 較低。但會學習兩種圖片中的風格，特徵等分布，所以 FID、KID、LPIPS 的指標反應出的結果會較優秀。

表 21. Cycle-GAN 訓練結果之評估分數

Cycle-GAN	
PSNR	9.812
SSIM	0.055
FID	8.999
KID	0.14
LPIPS	0.375

Cycle-GAN 訓練過程中生成圖片的情形如圖 24所示，在訓練中使用80000次的訓練次數，由圖可知在訓練第20000次的時候生成的圖片仍有一點模糊，直到訓練到80000次後生成器才有學習到轉換後真實房屋圖片的特徵，但是雖然學習到特徵並且生成的圖片清晰許多，但是在顏色上還是跟原圖有部分差異，這部分是因為 Cycle-GAN 並沒有使用像素損失，所以在學習上會盡量學習整個資料集中，顏色與結構的分布。在經過學習後Cycle-GAN 能夠學習到房屋中上面屋頂的部分與中間牆體的部分顏色會不同；另外也有學習到草圖中長方形部分應該要生成窗戶。所以生成圖片與真實圖片不相同，但在面對沒看過的資料也能根據草圖生成類似風格的房屋。模型的泛化性較高，也沒有出現過度擬合的問題。

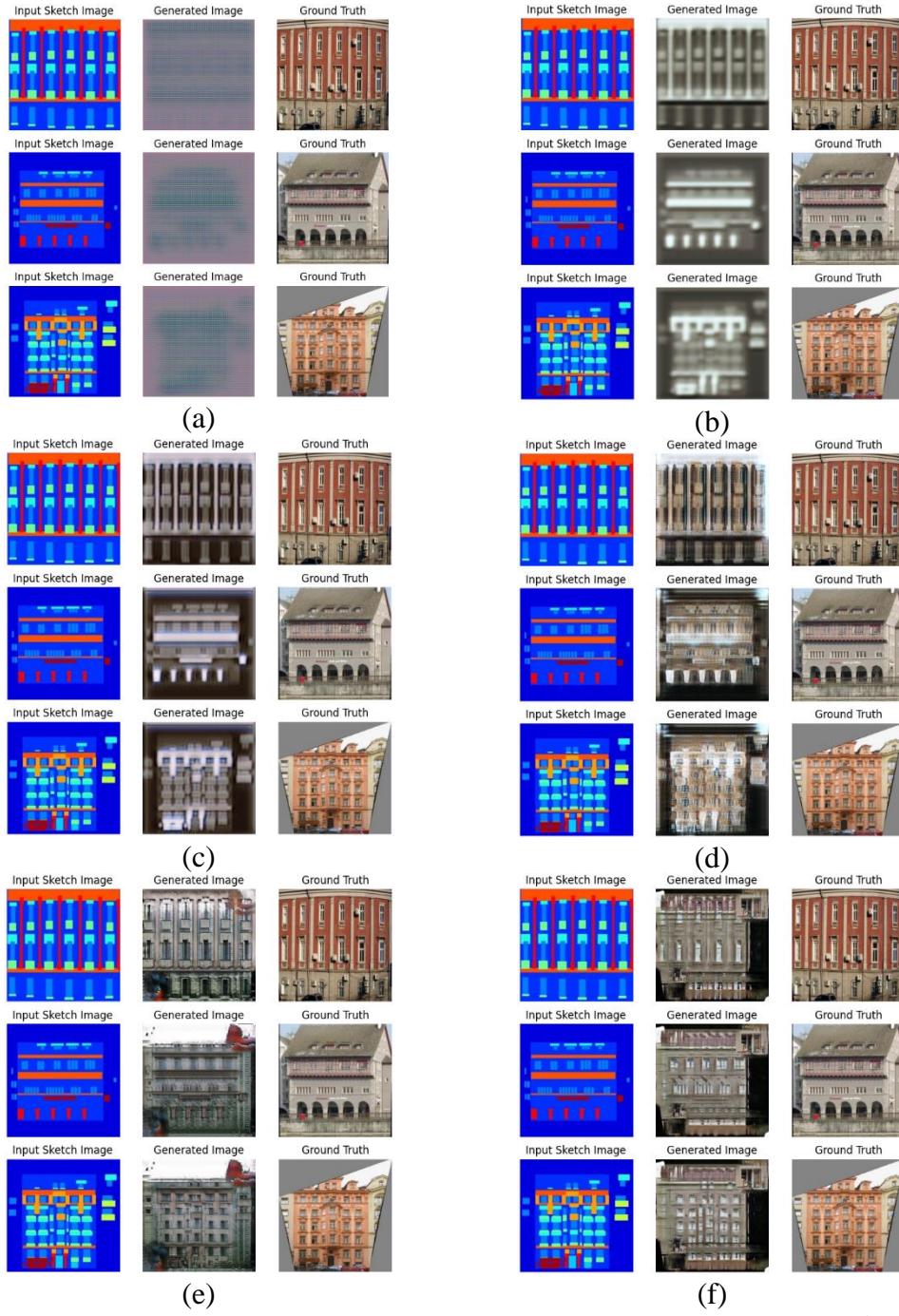


圖 24. 本研究建立之 Cycle-GAN 對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第200次訓練, (c) 第1000次訓練, (d) 第5000次訓練, (e) 第20000次訓練, (f) 第80000次訓練。

本研究中中的訓練損失如圖 25所示。其中又細分成三種訓練損失，分別是對抗損失、循環損失以及特徵損失。由圖可知雖然生成器與判別器是對抗式訓練，但是在訓練約10000次就趨於穩定，但是生成器的損失值還是

判別器高一點；而循環一致性損失以及特徵損失則用於生成器生成圖片的品質的損失計算。由圖可知循環一致性損失以及特徵損失也在訓練中期就收斂且逐漸趨於穩定，所以生成出來的圖片品質就並無太大的落差。

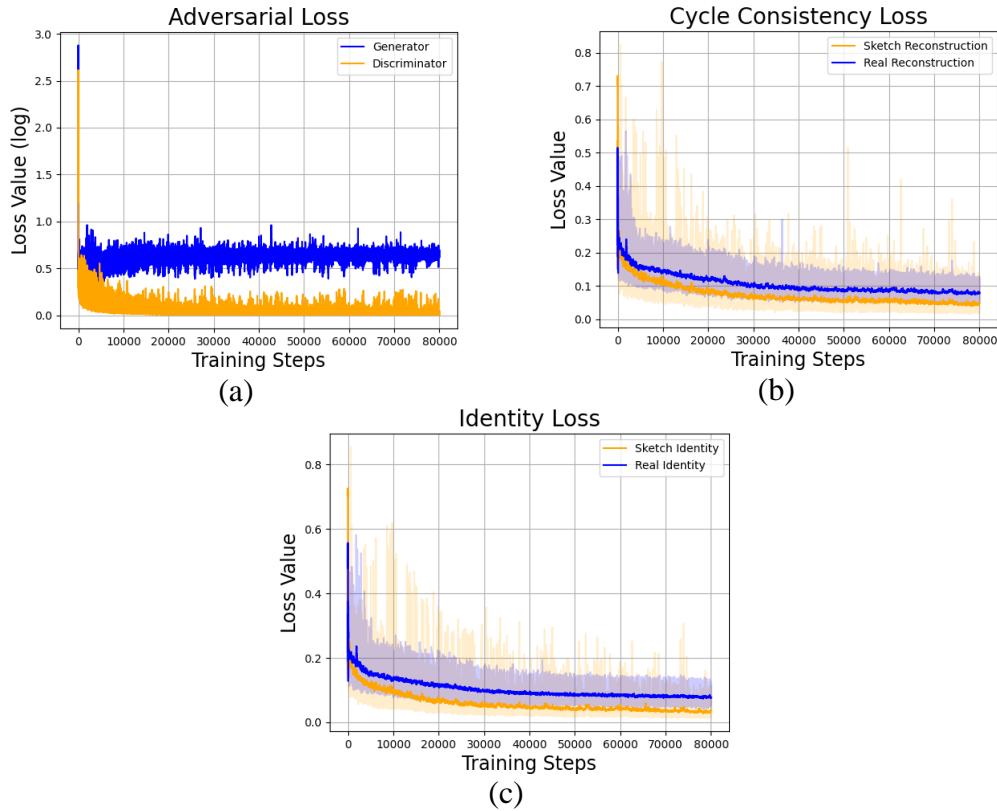


圖 25. Cycle-GAN 訓練損失, (a) 對抗損失, (b) 循環一致性損失, (c) 特徵損失。

對於 Cycle-GAN 測試資料之生成能力如圖 26所示，由圖可知生成出的圖片有學習到轉換後的圖片的特徵，像是顏色以及窗體的結構，還有房子建築上的架構，但仍有出現跟實際圖片些許不同之處。不過針對沒有學習過的測試資料集 Cycle-GAN 也能根據訓練時學習到的房屋特徵生成對應的圖片，也不會出現太多類似雜訊的瑕疵。

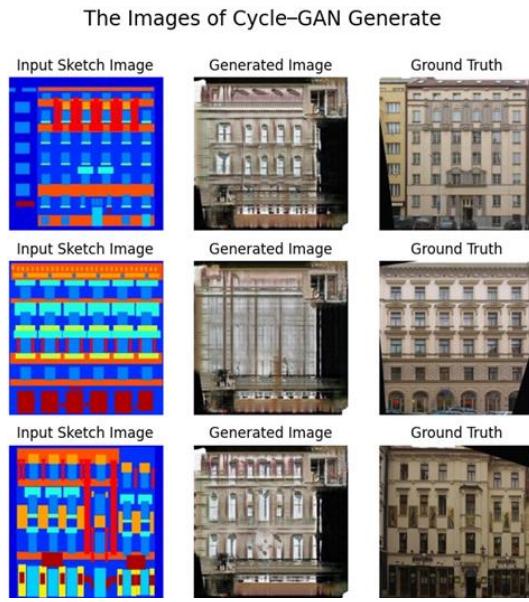


圖 26. Cycle-GAN 對於測試資料之生成能力

4.1.5 DDIM 生成結果探討

先前使用擴散模型訓練別的資料集時資料及數量都為50000筆到60000筆，而本研究之訓練資料集僅有400筆，就算使用資料擴增之後的2400筆資料似乎對於訓練的效果都不太好。另外訓練中發現使用解析度 256×256 的訓練資料對於運算的負擔太大，所以本研究也將資料解析度重整為 64×64 再進行訓練。目前在 256×256 解析度訓練的成果在一般無條件生成時就已經無法生成出良好的結果，訓練過程時所生成的結果如圖 27所示。未來會繼續探討模型架構對於訓練生成造成的影响並再加入條件控制。在 64×64 解析度訓練的成果比較好，但也因為此解析度較模糊所以生成結果仍然較模糊。雖然能夠大略學習到圖片特徵，但也無法非常良好的學習到圖片的特徵。在此解析度下的訓練過程如圖 28。

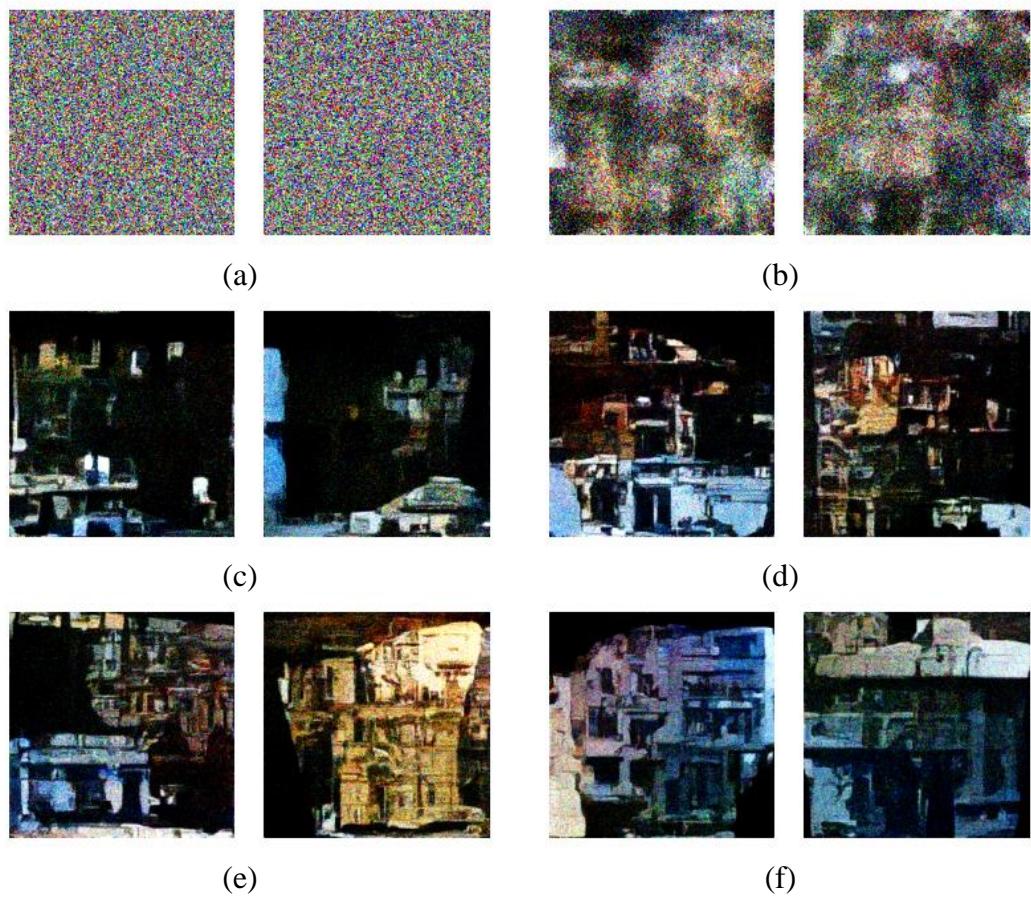


圖 27. 本研究 DDIM 在 256×256 解析度下對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第5次訓練, (c) 第15次訓練, (d) 第25次訓練, (e) 第35次訓練, (f) 第50次訓練。

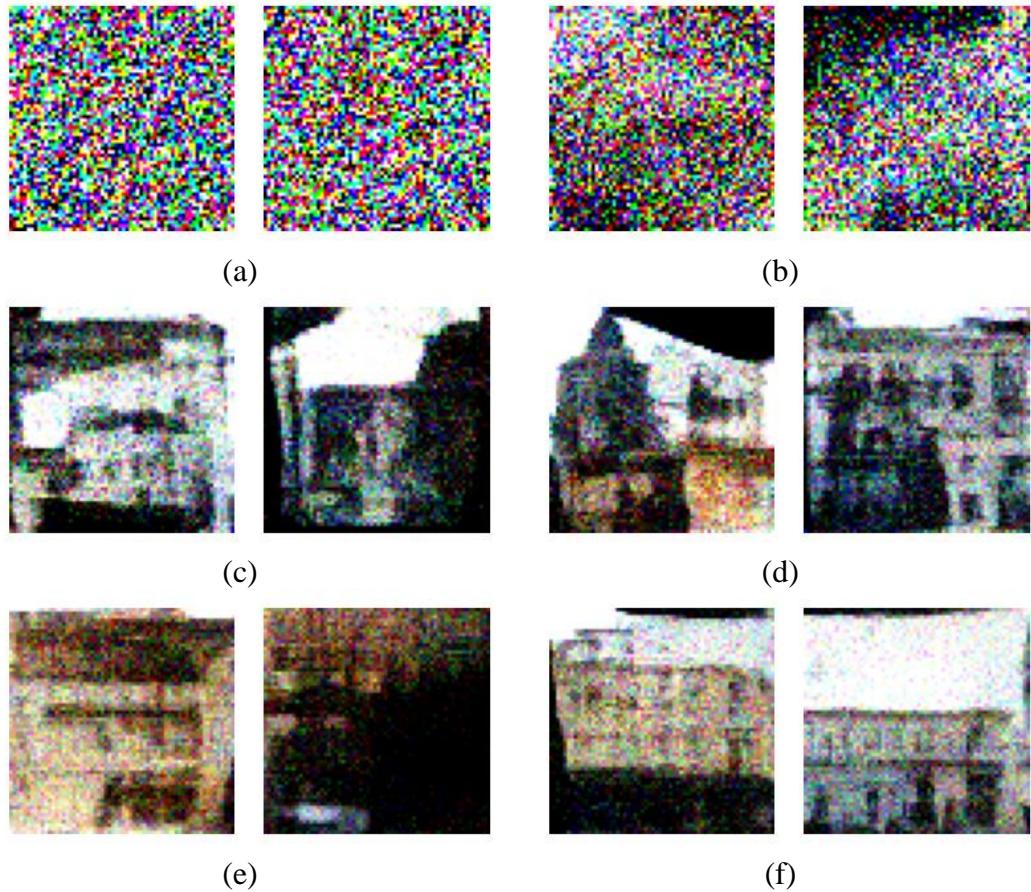


圖 28. 本研究 DDIM 在 64×64 解析度下對於訓練資料之訓練情形, (a) 第1次訓練, (b) 第5次訓練, (c) 第15次訓練, (d) 第25次訓練, (e) 第35次訓練, (f) 第50次訓練。

DDIM 訓練之損失函數變化如圖 29所示，其中 Noise Loss 曲線代表對於模型生成的雜訊損失、Image Loss 曲線為圖片生成之像素誤差、KID Score 曲線是對於驗證資料集的 KID 分數計算，訓練時模型只根據 Noise Loss 進行梯度計算與反向傳播。在 256×256 解析度訓練下，由圖可知 KID 分數呈現震盪不穩的狀態，但其他損失卻有收斂，現階段本研究認為生成模型可能已經學習到前向擴散之對應雜訊分布，但是在逆向擴散中因為沒有良好的學習到圖片的特徵等結果，所以在逆向擴散生成圖片時才會無法良好的生成正確的圖片。在 64×64 解析度訓練下，模型因為解析度變低，圖片特徵細節變少所以能夠較良好的生成圖片。損失與 KID 分數都有降低，訓練也有達到收斂。但也因為解析度本來就低，所以生成結果仍然比較模糊，圖片品質仍然較差。研究中我們認為在未來的研究中需要再注意資料

集的數量與品質，以訓練出較好的 DDIM 模型。

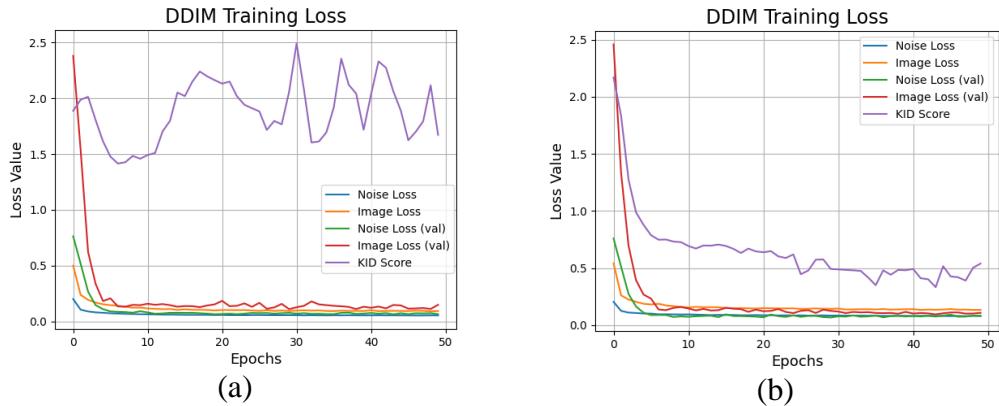


圖 29. 本研究 DDIM 在不同解析度訓練下之訓練損失變化, (a) 256×256 解析度, (b) 64×64解析度。

本研究 DDIM 生成圖片的結果如圖 30所示，可以看到256×256解析度下 DDIM 生成的圖片相當模糊，且圖片特徵的空間關係也相當混亂。而 64×64解析度的圖片則可以大致看出房子輪廓，但噪音沒有完全去除。也因為原始資料集也有一些遮蔽，所以在訓練過程中可能也將這些遮蔽當成圖片的一部分特徵，導致圖片邊緣會有一些遮蔽物存在。總得來說 DDIM 在訓練時需要高度依賴圖片的品質、資料集的數量等，故在未來使用 DDIM 訓練資料生成時，這些因素都需要慎重考慮。



圖 30. DDIM 對於不同解析度下測試資料之生成能力比較, (a) 256×256 解析度, (b) 64×64解析度。

4.2 模型生成結果比較

本研究在經過建構不同模型以及訓練後，對於所得之條件變換模型之生成結果比較如圖 31，由圖可知在經過條件輸入後由於 CVAE 與 Pix2Pix 在訓練時都有像素損失，所以對於已經學習過的資料能夠生成與真實圖幾

乎一致的圖片，但在面對沒有學習過的資料則 CVAE 較難生成可辨識的結果；而 Pix2Pix 則會根據訓練嘗試生成符合草圖條件的結果。CGAN 在訓練上則因模型概念較基礎，目標函數中並無特殊機制且生成器與判別器對抗訓練失衡而無法生成良好可辨識的圖片。而 Cycle-GAN 則因為模型訓練目標函數沒有生成圖片與真實圖片的像素損失，而是使用了循環一致性損失等機制導致模型可以學習到房屋的風格並利用草圖生成相應的風格。但也許因為訓練資料不足所以模型無法良好的學習到廣泛的房屋特徵表現，故生成結果可能偏向某些房屋特徵，不過細節卻可以因草圖而有不同。最後 DDIM 因未訓練其條件風格變換故無生成結果可供對照。

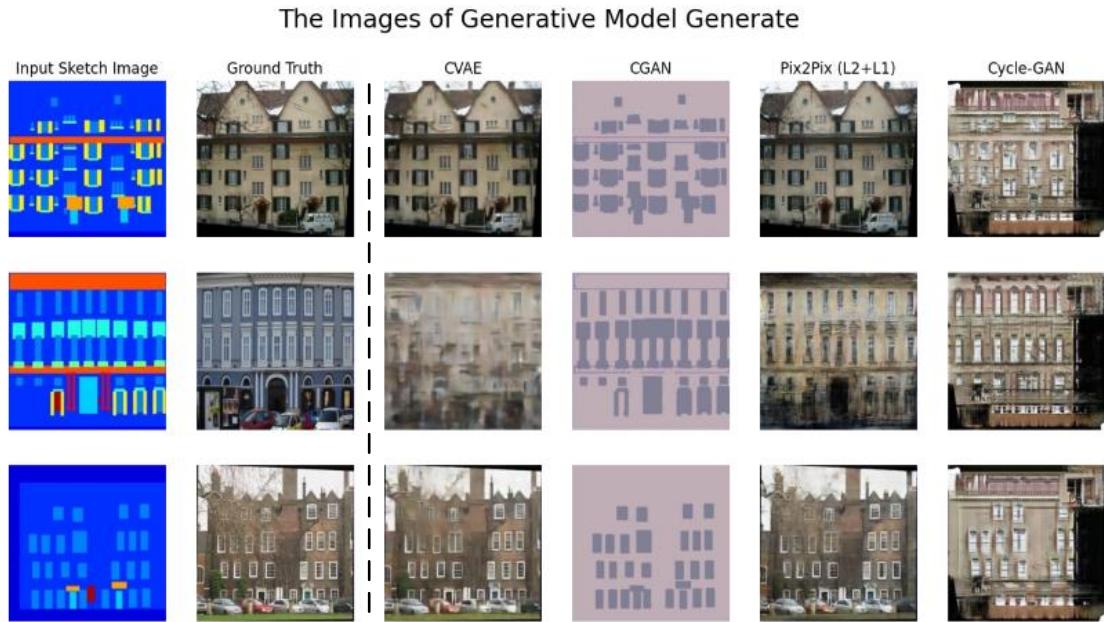


圖 31. 對於不同草圖輸入到不同模型之生成圖片比較

表 22為所有生成模型訓練完成後會隨機使用16張訓練集草圖與16張測試集草圖共32張草圖生成對應之房屋圖，以及對應之真實房屋圖去進行圖片相似度的計算，圖片皆為 256×256 解析度。DDIM 則在 64×64 解析度下直接生成32張圖片並隨機使用32張 64×64 解析度的真實房屋圖進行圖片相似度的計算。由表可知損失函數使用 L2+L1的 Pix2Pix 訓練指標皆為最佳的結果，其次在 KID 指標上 Cycle-GAN 為第二好的結果，分數為0.14僅次於損失函數使用 L2+L1的 Pix2Pix 的0.027，這代表 Cycle-GAN 生成的結果雖然與原圖不同但特徵上大致類似。不過在測試資料集的生成上 Cycle-GAN 與 Pix2Pix 生成的結果則為較容易辨識的，其他模型則因訓練中並未學習過資

料分布而無法生成較佳的結果。

表 22. 各生成模型訓練之評估分數

	CVAE	CGAN	Pix2Pix (L2+L1)	Pix2Pix (Bce+L1)	Cycle-GAN	DDIM (64px)
PSNR	14.387	7.492	23.725	13.675	9.812	2.346
SSIM	0.3	0.124	0.588	0.228	0.055	-0.004
FID	6.64	16.111	4.453	14.063	8.999	18.462
KID	0.227	0.443	0.027	0.329	0.14	0.264
LPIPS	0.355	0.696	0.186	0.431	0.375	0.439

第5章 結論與後續研究建議

5.1 結論

本專題使用多種生成模型來執行圖像轉圖像的圖像翻譯、風格變換類型任務，研究成果顯示在訓練中 Pix2Pix 與 CVAE 在經過訓練可以得到與訓練資料及幾乎一樣的圖片，但使用測試資料生成則圖片會出現明顯瑕疵，訓練上可能出現了過度擬合。CGAN 則完全無法生成可辨識的結果，模型生成結果幾乎都不可辨識，且訓練上還出現了梯度消失。而 Cycle-GAN 訓練後可以大致學習到訓練資料的一些表現特徵，並進行圖像風格的變換應用，且無論面對訓練資料抑或是測試資料都可以生成與草圖條件相應的結果。惟資料集數量較少故無法完整且廣泛地學習到圖片的細節特徵。

在需要大量運算資源的 DDIM 的訓練中，我們發現 DDIM 訓練時間比起其他模型來說更久。而且 DDIM 訓練上也很容易因為資料量不夠而無法良好的學習到圖片的特徵，即使使用資料增強對於訓練時資料量可能還略顯不足，未來訓練上則需要注意資料量是否充足。以及未來也需要著重研究 DDIM 在條件生成上的方式，何種方式能夠使擴散模型學習到更好的條件與對應風格之圖片生成，這都是未來研究值得探討的部分。

本專題的風格變換應用在未來也可以應用於許多不同的場合，例如在影像處理領域中可用於黑白圖片轉換為真實圖片、2D 圖片轉換為3D 圖片、將圖片轉換為短時間的影片等。在訊號處理領域也能根據不同風格的聲音、針對音樂等進行風格的變換。在機器人控制領域也可以用於機器人馬達控制的應用例如將機器人前進的腿部馬達控制參數改為向左走或者向右走、機器視覺轉換等應用。也能用於資料擴增等，例如分類任務中若某種類型資料不足也可以使用風格變換來補全資料不足的部分，對於一些項醫學影像等較難收集到的資料也有一定的幫助。總得來說資料風格變換的應用非常廣泛，在許多領域都能夠使用此技術來生成資料。

5.2 後續研究建議

因為此資料集的資料量不算太多，所以對於一些深層網路的模型，或者需要大量資料集以用於學習圖片細節特徵的模型在訓練上會有一些特徵

無法學習到。例如 DDIM 在訓練時效果就不太好，但在訓練有 50000 筆以上的資料集的任務時效果就不錯。故我們推測可能是訓練資料量不足以訓練擴散模型，且硬體設備效能不足，所以訓練 DDIM 時需要花費更多時間。未來可以使用具有更多資料的資料集、並且將電腦設備提升以用於良好的訓練 DDIM 類型的擴散模型。另外因為 CGAN 是較基礎的模型，所以應付此種較複雜的任務也比較吃力，未來就可以直接使用較複雜的模型例如 Cycle-GAN 甚至其他效能更強的模型進行風格變換的任務。

參考文獻

- [1] P.-H.Kuo andK.-L.Chen, “Two-stage fuzzy object grasping controller for a humanoid robot with proximal policy optimization,” *Eng. Appl. Artif. Intell.*, vol. 125, p. 106694, Oct.2023, doi: 10.1016/j.engappai.2023.106694.
- [2] Y.Tai, “A Survey Of Regression Algorithms And Connections With Deep Learning,” Apr.2021, [Online]. Available: <http://arxiv.org/abs/2104.12647>
- [3] A.Zell, G.Sumbul, andB.Demir, “Deep Metric Learning-Based Semi-Supervised Regression with Alternate Learning,” in *2022 IEEE International Conference on Image Processing (ICIP)*, IEEE, Oct. 2022, pp. 2411–2415. doi: 10.1109/ICIP46576.2022.9897939.
- [4] S.Roy *et al.*, “Deep Learning for Classification and Localization of COVID-19 Markers in Point-of-Care Lung Ultrasound,” *IEEE Trans. Med. Imaging*, vol. 39, no. 8, pp. 2676–2687, Aug.2020, doi: 10.1109/TMI.2020.2994459.
- [5] K.O’Shea andR.Nash, “An Introduction to Convolutional Neural Networks,” Nov.2015, [Online]. Available: <http://arxiv.org/abs/1511.08458>
- [6] P.Xuan, L.Gao, N.Sheng, T.Zhang, andT.Nakaguchi, “Graph Convolutional Autoencoder and Fully-Connected Autoencoder with Attention Mechanism Based Method for Predicting Drug-Disease Associations,” *IEEE J. Biomed. Heal. Informatics*, vol. 25, no. 5, pp. 1793–1804, May2021, doi: 10.1109/JBHI.2020.3039502.
- [7] D. P.Kingma andM.Welling, “Auto-Encoding Variational Bayes,” Dec.2013, [Online]. Available: <http://arxiv.org/abs/1312.6114>
- [8] I. J.Goodfellow *et al.*, “Generative Adversarial Networks,” Jun.2014, [Online]. Available: <http://arxiv.org/abs/1406.2661>
- [9] A.Radford, L.Metz, andS.Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” Nov.2015, [Online]. Available: <http://arxiv.org/abs/1511.06434>
- [10] M.Mirza andS.Osindero, “Conditional Generative Adversarial Nets,” Nov.2014, [Online]. Available: <http://arxiv.org/abs/1411.1784>
- [11] P.Isola, J.-Y.Zhu, T.Zhou, andA. A.Efros, “Image-to-Image Translation with Conditional Adversarial Networks,” Nov.2016, [Online]. Available: <http://arxiv.org/abs/1611.07004>
- [12] J.-Y.Zhu, T.Park, P.Isola, andA. A.Efros, “Unpaired Image-to-Image

- Translation using Cycle-Consistent Adversarial Networks,” Mar.2017, [Online]. Available: <http://arxiv.org/abs/1703.10593>
- [13] J.Ho, A.Jain, andP.Abbeel, “Denoising Diffusion Probabilistic Models,” Jun.2020, [Online]. Available: <http://arxiv.org/abs/2006.11239>
- [14] J.Song, C.Meng, andS.Ermon, “Denoising Diffusion Implicit Models,” Oct.2020, [Online]. Available: <http://arxiv.org/abs/2010.02502>
- [15] “Facades Dataset,” 2023. <https://www.kaggle.com/datasets/balraj98/facades-dataset> (accessed Sep.25, 2023).
- [16] D.Bank, N.Koenigstein, andR.Giryes, “Autoencoders,” Mar.2020, [Online]. Available: <http://arxiv.org/abs/2003.05991>
- [17] D. P.Kingma andJ.Ba, “Adam: A Method for Stochastic Optimization,” Dec.2014, [Online]. Available: <http://arxiv.org/abs/1412.6980>
- [18] “ImageNet,” 2023. <https://www.image-net.org/> (accessed Sep.23, 2023).
- [19] K.Muandet, K.Fukumizu, B.Sriperumbudur, andB.Schölkopf, “Kernel Mean Embedding of Distributions: A Review and Beyond,” *Found. Trends® Mach. Learn.*, vol. 10, no. 1–2, pp. 1–141, 2017, doi: 10.1561/2200000060.
- [20] R.Zhang, P.Isola, A. A.Efros, E.Shechtman, andO.Wang, “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric,” Jan.2018, [Online]. Available: <http://arxiv.org/abs/1801.03924>
- [21] “Variational Bayesian methods,” 2023. https://en.wikipedia.org/wiki/Variational_Bayesian_methods (accessed Sep.27, 2023).

誌謝

在研究中我們要感謝我們的指導教授，楊柏遠老師。老師在平常百忙之中還要抽空來了解進度與給予建議，在此獻上最高的謝意。另外也要感謝智慧機器人學系給予我們一間專題實驗室，讓我們可以在此完成研究。此外，我們還要感謝那些在專題研究過程中給予我們幫助、建議、指教的同學們。最後我們要感謝所有參與這項研究的人，感謝你們的貢獻和支持。

附錄

附錄1. 硬體設備

本研究使用之硬體設備如下表 23。

表 23. 研究使用之硬體設備資訊

設備名稱	設備資訊
CPU	intel (R) core (TM) i7-12700F
記憶體	32GB
GPU	NVIDIA GeForce RTX 3080 10GB

附錄2. 軟體環境

本研究中所使用之環境為 Python 3.8版本，整合開發環境 (Integrated Development Environment, IDE)為 Pycharm Professional 2021.1.2。研究中所使用到的套件版本如下表 24。

表 24. 研究使用之套件詳細資訊

套件名稱	版本編號
Numpy	1.19.2
Pandas	1.4.2
OpenCV	3.4.8.29
Keras	2.7.0
TensorFlow	2.7.0
TensorFlow-gpu	2.7.0
TensorFlow-addons	0.17.1
Matplotlib	3.5.2
Protobuf	3.20.3
Pytorch	2.1.0
Lpips	0.1.4

附錄3. 變分自動編碼器 (VAE)之原理推導

附錄3-1. VAE 訓練之 KL 散度計算

VAE 訓練之 KL 散度計算為了要使程式編輯較好撰寫所以將傳統 KL 散度的公式展開改寫。以便因應利用編碼器模型的輸出與平均為0、變異數為1的常態分布 $\mathcal{N}(0,1)$ 進行 KL 散度的計算。KL 散度的公式如(49)，其中 $p(x_n)$ 與 $q(x_n)$ 為兩個機率分布，其機率密度函數個別如(50)與(51)所示。

$$D_{KL}(p, q) = \sum p(x_n) \log \frac{p(x_n)}{q(x_n)} \quad (49)$$

$$p(x) = \frac{1}{\sqrt{2\pi\sigma_p^2}} e^{-\frac{(x-\mu_p)^2}{2\sigma_p^2}} \quad (50)$$

$$q(x) = \frac{1}{\sqrt{2\pi\sigma_q^2}} e^{-\frac{(x-\mu_q)^2}{2\sigma_q^2}} \quad (51)$$

VAE 要計算編碼器輸出與 $\mathcal{N}(0,1)$ 的距離，也就是他們的 KL 散度。

由(49)式利用對數律展開可得到(52)。

$$D_{KL}(p, q) = \sum [p(x)(\log p(x) - \log q(x))] \quad (52)$$

接著本研究探討分布連續的情況下，所以將方程式改為積分型式，故上式首先將 $p(x)$ 帶入並分成兩個項目，可以得到(53)。

$$D_{KL}(p, q) = - \int p(x) \log q(x) dx + \int p(x) \log p(x) dx \quad (53)$$

接著此式分為第一項 $-\int p(x) \log q(x) dx$ 與第二項 $\int p(x) \log p(x) dx$ ，接下來將分開計算。首先探討第一項的化簡，可以將 $q(x)$ 使用機率密度函數(51)的方式帶入，則該項會變成(54)。

$$-\int p(x) \log q(x) dx = - \int p(x) \log \frac{1}{\sqrt{2\pi\sigma_q^2}} e^{-\frac{(x-\mu_q)^2}{2\sigma_q^2}} dx \quad (54)$$

使用對數律將 \log 中相乘的項目變為兩個 \log 函數相加，接下來再使用對數律化簡(55)， \log 與 e 又可以消除； \log 中的根號($\frac{1}{2}$ 次方)可以移至 \log 外面(56)。

$$-\int p(x) \log q(x) dx = - \int p(x) \left[\log e^{-\frac{(x-\mu_q)^2}{2\sigma_q^2}} - \log \sqrt{2\pi\sigma_q^2} \right] dx \quad (55)$$

$$-\int p(x) \log q(x) dx = \frac{1}{2} \log(2\pi\sigma_q^2) + \int p(x) \frac{(x-\mu_q)^2}{2\sigma_q^2} dx \quad (56)$$

將 $(x-\mu_q)^2$ 展開並將 $p(x)$ 乘入分子項，並將分子項目分開進行積分計算。若有變數 x 則 x 與 x^2 的期望值可用(58)與(59)表示。

$$-\int p(x) \log q(x) dx = \frac{1}{2} \log(2\pi\sigma_q^2) + \frac{\int x^2 p(x) dx - \int 2x\mu_q p(x) dx + \int \mu_q^2 p(x) dx}{2\sigma_q^2} \quad (57)$$

$$\mathbb{E}[x] = \int xp(x) dx = \mu_p \quad (58)$$

$$\mathbb{E}[x^2] = \int x^2 p(x) dx = \sigma_p^2 + \mu_p^2 \quad (59)$$

將(58)與(59)帶入(57)，則方程式會變成(60)，再化簡會變成(61)。

$$-\int p(x) \log q(x) dx = \frac{1}{2} \log(2\pi\sigma_q^2) + \frac{\sigma_p^2 + \mu_p^2 - 2\mu_p\mu_q + \mu_q^2}{2\sigma_q^2} \quad (60)$$

$$-\int p(x) \log q(x) dx = \frac{1}{2} \log(2\pi\sigma_q^2) + \frac{\sigma_p^2 + (\mu_p - \mu_q)^2}{2\sigma_q^2} \quad (61)$$

接下來要計算 $\int p(x) \log p(x) dx$ 的化簡，流程與 $-\int p(x) \log q(x) dx$ 一樣，故接下來將省略說明直接將計算過程推導一次，經過(62)的計算可得到(63)。

$$\begin{aligned} \int p(x) \log p(x) dx &= \int p(x) \log \frac{1}{\sqrt{2\pi\sigma_p^2}} e^{-\frac{(x-\mu_p)^2}{2\sigma_p^2}} dx \\ &= \int p(x) \left[\log e^{-\frac{(x-\mu_p)^2}{2\sigma_p^2}} - \frac{1}{2} \log(2\pi\sigma_q^2) \right] dx \\ &= -\frac{1}{2} \log(2\pi\sigma_p^2) + \int p(x) \frac{-(x-\mu_p)^2}{2\sigma_p^2} dx \\ &= -\frac{1}{2} \log(2\pi\sigma_p^2) - \frac{\int x^2 p(x) dx - \int 2x\mu_p p(x) dx + \int \mu_p^2 p(x) dx}{2\sigma_p^2} \end{aligned} \quad (62)$$

$$\begin{aligned} &= -\frac{1}{2} \log(2\pi\sigma_p^2) - \frac{\mathbb{E}[x^2] - 2\mathbb{E}[x]\mu_p + \mu_p^2}{2\sigma_p^2} \\ &= -\frac{1}{2} \log(2\pi\sigma_p^2) - \frac{\sigma_p^2 + \mu_p^2 - 2\mu_p^2 + \mu_p^2}{2\sigma_p^2} \\ \int p(x) \log p(x) dx &= -\frac{1}{2} \log(2\pi\sigma_p^2) - \frac{1}{2} \end{aligned} \quad (63)$$

接著將(61)跟(63)帶回(53)，可以得到(64)。

$$D_{KL}(p, q) = \frac{1}{2} \log(2\pi\sigma_q^2) - \frac{1}{2} \log(2\pi\sigma_p^2) - \frac{1}{2} + \frac{1}{2} \left[\frac{\sigma_p^2}{\sigma_q^2} + \frac{(\mu_p - \mu_q)^2}{\sigma_q^2} \right] \quad (64)$$

將 $\frac{1}{2}$ 提出來後第一項與第二項由對數律合併可得到(65)，再將方程式乘以(-1)即可得到 $D_{KL}(p, q)$ 的展開式(66)。

$$D_{KL}(p, q) = \frac{1}{2} \left(\log \frac{2\pi\sigma_q^2}{2\pi\sigma_p^2} - 1 + \frac{\sigma_p^2}{\sigma_q^2} + \frac{(\mu_p - \mu_q)^2}{\sigma_q^2} \right) \quad (65)$$

$$D_{KL}(p, q) = -\frac{1}{2} \left(1 - \log \frac{\sigma_q^2}{\sigma_p^2} - \frac{\sigma_p^2}{\sigma_q^2} - \frac{(\mu_p - \mu_q)^2}{\sigma_q^2} \right) \quad (66)$$

最後由於本研究要計算 $D_{KL}(p, \mathcal{N}(0,1))$ ，故將常態分布 $\mathcal{N}(0,1)$ 之平均 $\mu_q = 0$ 與變異數 $\sigma_q^2 = 1$ 帶入公式(66)，即可得到(67)。

$$\begin{aligned} D_{KL}(p, \mathcal{N}(0,1)) &= -\frac{1}{2} \left(1 - \log \frac{1}{\sigma_p^2} - \frac{\sigma_p^2}{1} - \frac{\mu_p^2}{1} \right) \\ &= -\frac{1}{2} (1 + \log \sigma_p^2 - \sigma_p^2 - \mu_p^2) \end{aligned} \quad (67)$$

此公式即為用於計算編碼器輸出之編碼與常態分布的差距，使用此公式能於程式碼中更簡單的定義損失。其中 $\log \sigma_p^2$ 為編碼器輸出的其中一部分 $\log var$ ； μ_p^2 為編碼器輸出的另一個部分 $mean$ ，輸出的演算如 Algorithm 1。 σ_p^2 就是 $\log \sigma_p^2$ 取 e 的 $\log \sigma_p^2$ 次方的結果，如(68)。

$$\sigma_p^2 = e^{\log \sigma_p^2} \quad (68)$$

附錄3-2. VAE 訓練之目標函數推導

VAE 訓練目標之完整推導過程較複雜，主要是使用變分下界 (Variational Lower Bound, VLB)，也稱 ELBO。優化負對數似然 (Negative Log Likelihood, NLL)。也是將 KL 散度給改寫，首先定義兩個條件分布的 KL 散度。

$$D_{KL}(q_\phi(z|x)||p_\theta(z|x)) = \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(z|x)} dz \quad (69)$$

於是可以在將 $p_\theta(z|x)$ 使用條件機率 (70) 替換並改寫成 (71)。

$$p_\theta(z|x) = \frac{p(z,x)}{p(x)} \quad (70)$$

$$D_{KL}(q_\phi(z|x)||p_\theta(z|x)) = \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(z,x)} p(x) dz \quad (71)$$

接著利用對數率將真數裡面相乘的部分拉到對數函數外變成相加。

$$\int q_\phi(z|x) \left(\log \frac{q_\phi(z|x)}{p_\theta(z,x)} + \log p_\theta(x) \right) dz \quad (72)$$

然後將括號展開，接著因為 (73)，所以方程可以計算成 (74)：

$$\int q_\phi(z|x) \log p(x) dz = \log p(x) \quad (73)$$

$$\log p_\theta(x) + \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(z|x)} dz \quad (74)$$

再來使用貝氏定理 (75) 改寫方程。

$$p_\theta(z|x) = p(x|z)p(z) \quad (75)$$

$$\log p_\theta(x) + \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(x|z)p_\theta(z)} dz \quad (76)$$

使用對數率將分母的項次獨立出來，並且改寫成期望值之形式。

$$\log p_\theta(x) + \mathbb{E}_{z \sim q_\phi(z|x)} \left[\log \frac{q_\phi(z|x)}{p_\theta(z)} - \log p_\theta(x|z) \right] \quad (77)$$

將期望值中第一項改寫成 KL 散度之形式，於是可以在原式改寫成

$$D_{KL}(q_\phi(z|x) || p_\theta(z|x)) = \log p_\theta(x) + D_{KL}(q_\phi(z|x) || p_\theta(z)) - \mathbb{E}_{z \sim q_\phi(z|x)} [\log p_\theta(x|z)] \quad (78)$$

接著移項。

$$\mathbb{E}_{z \sim q_\phi(z|x)} [\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x) || p_\theta(z)) = \log p_\theta(x) - D_{KL}(q_\phi(z|x) || p_\theta(z|x)) \quad (79)$$

上述方程式中，對於訓練的目標，需要最大化生成資料分布的對數

似然 ($\log p_\theta(x)$)，而且還要最小化真實資料與生成之分布的差異。

所以損失可以定義成下式：

$$L(\theta, \phi) = -\log p_\theta(x) + D_{KL}(q_\phi(z|x) || p_\theta(z|x)) \quad (80)$$

在變分貝葉斯方法 (Variational Bayesian methods)[21] 中，這種損失函數被稱為變分下界或者證據下界 (evidence lower bound, ELBO)，方程式的 KL 散度會大於 0，所以 $-L$ 是下界 $\log p_\theta(x)$ 。

$$-L = \log p_\theta(x) - D_{KL}(q_\phi(z|x) || p_\theta(z|x)) \leq \log p_\theta(x) \quad (81)$$

附錄4. 擴散模型之原理推導

附錄4-1. 推導前向擴散之原始圖片與雜訊圖片之關係

前向擴散時有公式 (17) 和 (18)，由此可以得知圖片 x_t 的計算方式如下：

$$x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} z_t \quad (82)$$

接著論文作者定義了 $\alpha_t = 1 - \beta_t$ 用於代替 β_t ，所以原式變為：

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1 - \alpha_t} z_{t-1} \quad (83)$$

$$x_{t-1} = \sqrt{\alpha_{t-1}} x_{t-2} + \sqrt{1 - \alpha_{t-1}} z_{t-2} \quad (84)$$

將方程式 (84) 帶入 (83)，可以得到方程式：

$$x_t = \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}} z_{t-2} \quad (85)$$

所以根據此規律可以算出 x_t 與 x_0 的關係，我們將 $\alpha_t \alpha_{t-1} \dots \alpha_0$ 表達如下：

$$\bar{\alpha}_t = \prod_{s=1}^t \alpha_s \quad (86)$$

將此方程帶入則 (85) 會變為 (87)，所以可進一步表達成常態分布的形式 (88)。

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} z \quad (87)$$

$$q(x_t | x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, 1 - \bar{\alpha}_t I) \quad (88)$$

接著移項計算 x_0 與 x_t 的關係，可以得知：

$$x_0 = \frac{x_t - \sqrt{1 - \bar{\alpha}_t} z}{\sqrt{\bar{\alpha}_t}} \quad (89)$$

附錄4-2. 推導目標函數

目標函數與 VAE 類似，也是使用 ELBO 來優化 NLL，優化計算結束後會得到 (90)，計算過程如附錄3-2。

$$-\log p_\theta(x_0) + D_{KL}(q(x_{1:T}|x_0) \| p_\theta(x_{1:T}|x_0)) \geq -\log p_\theta(x_0) \quad (90)$$

再來將神經網路訓練出來的分布 $p_\theta(x_{1:T}|x_0)$ 使用條件機率的公式計算成 $\frac{p_\theta(x_{0:T})}{p_\theta(x_0)}$ 接著再將 KL 散度 (49) 的公式展開。於是得到：

$$-\log p_\theta(x_0) \leq -\log p_\theta(x_0) + \mathbb{E}_q \left[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})} + \log p_\theta(x_0) \right] \quad (91)$$

接著 $\log p_\theta(x_0)$ 與 $-\log p_\theta(x_0)$ 抵消。

$$\mathbb{E}_q[-\log p_\theta(x_0)] \leq \mathbb{E}_q \left[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})} \right] \quad (92)$$

再將不等號右邊的項目展開，於是得到：

$$\mathbb{E}_q \left[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})} \right] = \mathbb{E}_q \left[\log \frac{\prod_{t=1}^T q(x_t|x_{t-1})}{p_\theta(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t)} \right] \quad (93)$$

把將分母的 $p_\theta(x_T)$ 移至外面：

$$\mathbb{E}_q \left[-\log p_\theta(x_T) + \log \frac{\prod_{t=1}^T q(x_t|x_{t-1})}{\prod_{t=1}^T p_\theta(x_{t-1}|x_t)} \right] \quad (94)$$

根據對數率，指數函數內的真數相乘可以視為不同指數函數函數相加：

$$\mathbb{E}_q \left[-\log p_\theta(x_T) + \sum_{t=1}^T \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)} \right] \quad (95)$$

於是可以在擴散時間 $t=1$ 的部分另外獨立。

$$\mathbb{E}_q \left[-\log p_\theta(x_T) + \sum_{t>1}^T \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)} + \log \frac{(x_1|x_0)}{p_\theta(x_0|x_1)} \right] \quad (96)$$

然後我們將連加符號內的前向擴散之分布加入條件 x_0 並使用貝氏定理將結果展開：

$$\mathbb{E}_q \left[-\log p_\theta(x_T) + \sum_{t>1}^T \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} \frac{q(x_t|x_0)}{q(x_{t-1}|x_0)} + \log \frac{(x_1|x_0)}{p_\theta(x_0|x_1)} \right] \quad (97)$$

接著也把 $\frac{q(x_t|x_0)}{q(x_{t-1}|x_0)}$ 獨立出來。

$$\mathbb{E}_q \left[-\log p_\theta(x_T) + \sum_{t>1} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} + \sum_{t>1} \frac{q(x_t|x_0)}{q(x_{t-1}|x_0)} + \log \frac{(x_1|x_0)}{p_\theta(x_0|x_1)} \right] \quad (98)$$

我們將第四項中的結果進行馬可夫鏈的計算，算出成果後如(99)式

$$\sum_{t>1} \frac{q(x_t|x_0)}{q(x_{t-1}|x_0)} = \log \frac{q(x_T|x_0)}{q(x_1|x_0)} \quad (99)$$

根據對數率，指數函數內的真數相除可以視為不同指數函數相減，然後再消除重複的項目後會變成：

$$\mathbb{E}_q = \left[\log q(x_T|x_0) - \log p_\theta(x_T) + \sum_{t>1} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} - \log p_\theta(x_0|x_1) \right] \quad (100)$$

於是再使用對數率將第一項與第二項合併，之後可以化成 KL 散度的形式，於是目標函數變成 (101)，這也是原始論文中提出的公式。不過因為 L_T 無訓練參數，所以計算出來是常數，可以忽略； L_0 為重構項目，計算出的值非常小，可以忽略。故接著要將 L_{t-1} 重參數化，求得最終損失。

$$\mathbb{E}_q [L_T + L_{t-1} + L_0] \quad (101)$$

$$L_T = D_{KL}(q(x_T|x_0) || p_\theta(x_T)) \quad (102)$$

$$L_{t-1} = \sum_{t>1} D_{KL}(q(x_{t-1}|x_t, x_0) || p_\theta(x_{t-1}|x_t)) \quad (103)$$

$$L_0 = -\log p_\theta(x_0|x_1) \quad (104)$$

附錄4-3. 推導 $q(x_{t-1}|x_t, x_0)$ 之分布平均與變異數

要求得 $q(x_{t-1}|x_t, x_0)$ 的後驗機率分布，我們根據論文原文假設 (105)。

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I) \quad (105)$$

根據貝氏定理， $q(x_{t-1}|x_t, x_0)$ 正比於下式。

$$\exp\left(-\frac{1}{2}\left(\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{\beta_t} + \frac{(x_{t-1} - \sqrt{\bar{\alpha}_{t-1}}x_0)^2}{1 - \bar{\alpha}_{t-1}} - \frac{(x_t - \sqrt{\bar{\alpha}_t}x_0)^2}{1 - \bar{\alpha}_t}\right)\right) \quad (106)$$

然後將方程內的項目都展開，並結合在一起後，我們將方程使用分配率整理成與 x_{t-1} 相關的方程式，因為 x_t 與 x_0 並不會用到故整理成一個函數 C 用於表示。

$$\exp\left(-\frac{1}{2}\left[\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1 - \bar{\alpha}_{t-1}}\right)x_{t-1}^2 - \left(\frac{2\sqrt{\alpha_t}}{\beta_t}x_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1 - \bar{\alpha}_{t-1}}x_0\right)x_{t-1} + C(x_t, x_0)\right]\right) \quad (107)$$

根據機率密度函數可以知道變異數 $\tilde{\beta}_t$ 等於 x_{t-1}^2 項常數的倒數 (108)，平均是 x_{t-1} 項常數除以 x_{t-1}^2 項常數的結果 (109)。

$$\tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t \quad (108)$$

$$\tilde{\mu}_t = \left(\frac{\sqrt{\alpha_t}}{\beta_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1 - \bar{\alpha}_{t-1}}x_0\right) \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t \quad (109)$$

再將 (89) 帶入至 (109) 可以得到欲求分布之平均的表達式

$$\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} z \right) \quad (110)$$

附錄4-4. 推導擴散模型之損失函數

根據文獻，擴散模型之損失函數如 (21)、模型訓練的逆向擴散過程之分布如 (20)。模型要使預測分布的平均 μ_θ 逼近 $\tilde{\mu}_t$ ，故 μ_θ 可表達成下式。

$$\mu_\theta = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} z_\theta(x_t, t) \right) \quad (111)$$

其中 x_t 與 t 都是訓練之輸入資料，分別代表雜訊圖與擴散時間。因此可以計算 μ_θ 與 $\tilde{\mu}_t$ 的誤差，我們使用 MSE，其中 C 為常數：

$$L_{t-1} - C = \mathbb{E}_{x_0, z} \left[\frac{1}{2\sigma_t^2} \|\tilde{\mu}_t(x_t | x_0) - \mu_\theta(x_t, t)\|^2 \right] \quad (112)$$

$$L_{t-1} - C = \mathbb{E}_{x_0, z} \left[\frac{1}{2\sigma_t^2} \left\| \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} z \right) - \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} z_\theta(x_t, t) \right) \right\|^2 \right] \quad (113)$$

使用分配率合併公式中的項目。先將 $\frac{1}{\sqrt{\alpha_t}}$ 後面的部分合併再將 $\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}$ 提出來，所以公式會變成：

$$L_{t-1} - C = \mathbb{E}_{x_0, z} \left[\frac{\beta_t^2}{2\alpha_t(1-\bar{\alpha}_t)\sigma_t^2} \|(z - z_\theta(x_t, t))\|^2 \right] \quad (114)$$

最後根據論文作者所述，將前面常數項移除可以得到比較好的結果；然後把 (87) 帶入至 (114)。故最後的損失函數會變成如下所示，也是公式 (21) 的完整推導過程。

$$L_{simple}(\theta) = \mathbb{E}_{t, x_0, z} \left[\|z - z_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z, t)\|^2 \right] \quad (115)$$