**Project Proposal**

**Neural Network Transfer on Application Icons**

110550174鄧人豪

110550159劉秉庭

**Problem Statement and Task Definition**

　　在電腦、手機上安裝圖示包的時候，圖示包常常不會支援所有的應用程式，所以就變得有些程式的圖示風格與其他程式不一致，如下圖所示：



我不禁想，這些主題怎麼不支援所有程式再發布，但是在深入了解圖示主題的設計後，發現原來一個圖示主題要名確的重新定義每個應用程式的圖示，也就是要包含幾個程式就要畫幾張圖，對圖示主題的設計者來說，要包含所有應用程式是不可能的。於是我們想讓電腦能根據設計師已經設計好的圖示來自動推斷不支援的應用程式圖示應該長怎樣，讓電腦上的圖示風格保持一致，增加美感。

　　我們最終希望能設計一個程式，檢測電腦目前正在使用的圖示主題，並看電腦上有哪些程式不被當前的圖示主題支援，利用 AI 模型將這些程式的圖示轉換成與當前圖示主題的設計風格、哲學相符的圖片。

**Description of the challenges**

　　這個問題有幾個難點需要解決。首先，經過初步搜尋後，似乎沒有現成的 dataset 適合用於本研究。好在圖示主題都是已經數位化且程式可讀的格式，不用手動標記。我們打算寫個腳本來自動化建立 dataset 的部份，假設資料不足，預計使用Data augmentation增加dataset size。

　　此外，圖示檔大多都是向量圖的格式，為了方便模型訓練，我們會需要將向量圖轉換為點陣圖，模型的輸出也會是點陣圖。但是這又會延伸出另一個問題，模型若輸出點陣圖，在高解析度的螢幕中看起來就會比較不清楚，但我們暫時決定先透過足夠高解析度的點陣圖來解決這個問題，若有時間再思考如何讓模型的輸入和輸出都是向量圖。

　　而想要做到設計風格的轉換，不單單只是改變顏色和筆刷的樣式，模型需要能學習到設計師的設計理念，再生成出相應的圖片，這與傳統的影像風格轉換有些不同。因此我們這次打算用CycleGAN來嘗試。

　　最後，我們希望模型能學到「學習圖示設計風格與產出圖示的能力」，讓我們能給他在訓練時沒看過的圖示主題，吸收該主題的設計風格，並套用到新的圖示上。不過這部分難度較高，我們會再視情況而定。

**Input/Output Behavior with Concrete Examples.**

　　輸入與輸出的範例如下表所示，我們會先告訴模型，當前的圖示主題定義的應用程式圖示 (Style Icon) 與這些應用程式原本的圖示 (Default Icons) 以及一個不在當前主題中的程式圖示(Input Image)，模型要根據 Default Icons 和 Style Icons 的關係來推斷出 Input Icon 會變成怎樣後，輸出 Output Icon，輸入及輸出範例如下(表一)。

(表一) 輸入及輸出範例

|  |  |  |  |
| --- | --- | --- | --- |
| **⋮** | **⋮** |  |  |
| Default Icons | Style Icons | Input Icon | Output Icon |
| Input Image | | | Output Image |

其中圖片都會是以 512x512 的 np.array 表示，若檔案的格式原本是 svg 向量圖檔，會被轉成 512x512 的點陣圖來操作。

**Related works**

　　在2017的**Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks [1]**中，他們使用CycleGAN對圖片的風格及內容進行轉換，包括斑馬與馬、畫與真實圖片、冬天與夏天等等的互相轉換。從那時開始，有許多人嘗試用CycleGAN進行圖片的轉換，例如：**Van Huy**使用CycleGAN將不同水果互相轉換 **[2]**；**Yanghua JIN**使用CycleGAN將圖片中角色的髮色都轉為銀色 **[3]** 等等。

**Methodology**

　　由於我們想要做到設計風格的轉換，不單單只是改變顏色和筆刷的樣式，因此會嘗試實作CycleGAN來達成，實作方法大致可依照**Understanding and Implementing CycleGAN in TensorFlow [4]** 這篇文章中的方法。主要用Encoder, Transformer, Decoder來組成Generator，再用Convolutional layer組成Discriminator；接著分別計算Discriminator Loss, Generator Loss 與CycleGAN特有的Cyclic Loss。最後利用這些來建構模型，期望能夠順利轉換設計風格。不過實作之後有可能會發現圖示的形狀改變仍然太小，這時或許就需要參考在**Improving Shape Deformation in Unsupervised Image-to-Image Translation [5]** 中所使用的方法，讓轉換後的風格更接近目標的設計風格。

**Evaluation Metrics**

　　在使用GAN的時候，最基本需要觀察的應該就是各個Loss隨著epoch數增加所顯示的情況，而在CycleGAN中，總共有三種重要的Loss，分別是Generator Loss, Discrimination Loss和Cyclic Loss，因此我們將以epoch數為x軸、Losses 為y軸繪製圖表觀察。接著，在GAN中，根據應用種類產生非常多種的evaluation method，不過在CycleGAN中，較適合我們使用的應是**Fréchet Inception Distance(FID)**，因為它可以代表創造出來的fake image 與real image 相比的Fidelity和Diversity，這剛好是我們將圖示轉換風格後需要觀察的，因此選擇了FID。

**Baselines**說到風格轉換，不可忽略的另外一種方法就是Neural Style Transfer(NST)，在輸入內容影像與風格影像各一張後，可以生成一個新的風格化影像。

　　自從2015的**A Neural Algorithm of Artistic Style [6]** 開始使用VGG模型提取影像的特徵值以及緊接著的**Image Style Transfer Using Convolutional Neural Networks [7]** 之後，NST就被大眾使用，運用CNN之中的layer取得特徵值藉以進行風格轉換。

　　而現今的NST大多使用了Pretrained model，因此在擷取影像的特徵值部分不需要多餘的影像進行訓練，只需要Content image(內容影像)與Style image(風格影像)各一張輸入模型即可，最終會輸出一張新的Stylized image(風格化影像)，而我們在使用VGG19進行實作之後，NST的輸入及輸出影像範例如下(表二)：

(表二) NST輸入及輸出範例

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Input Image | | Output Image |
| Content Image | Style Image | Stylized Image |

　　可以看到，雖然輸出影像看似成功轉換風格，但這並不完全符合我們的預期，因為我們想要做到設計風格的轉換，不單單只是改變顏色和筆刷的樣式，模型需要能學習到設計師的設計理念，再生成出相應的圖片，這與NST的應用結果有些不同，因此我們單純使用NST作為Baseline。

**Time Schedule & Discussion**

紀錄於HackMD Note: <https://hackmd.io/hqtw6dL1QqO0-Tpk7wyR6g?fbclid=IwAR3LIbQ2jg6Oxp6ne3xd4FAZO9ZyypL6Kguu-iN1LiTIq3goRWjyBm5cqyM#Evaluation-Metrics>

**Repo**

<https://github.com/DENGRENHAO/AI_Project>

**References**

[1] Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. arXiv: 1703.10593, 2017

[2] Van Huy. CycleGAN-TensorFlow(<https://github.com/vanhuyz/CycleGAN-TensorFlow>). 2021

[3] Yanghua JIN. chainer-cyclegan(<https://github.com/Aixile/chainer-cyclegan>).

[4] Hardik Bansal, Archit Rathore. Understanding and Implementing CycleGAN in TensorFlow(<https://hardikbansal.github.io/CycleGANBlog/>).  
[5] Aaron Gokaslan, Vivek Ramanujan, Daniel Ritchie, Kwang In Kim, James Tompkin. Improving Shape Deformation in Unsupervised Image-to-Image Translation. arXiv: 1808.04325, 2018

[6] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge. A Neural Algorithm of Artistic Style. arXiv:1508.06576, 2015

[7] L. A. Gatys, A. S. Ecker and M. Bethge, "Image Style Transfer Using Convolutional Neural Networks," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2414-2423, doi: 10.1109/CVPR.2016.265.