# 111.1 Multimedia Information Systems Final presentation report

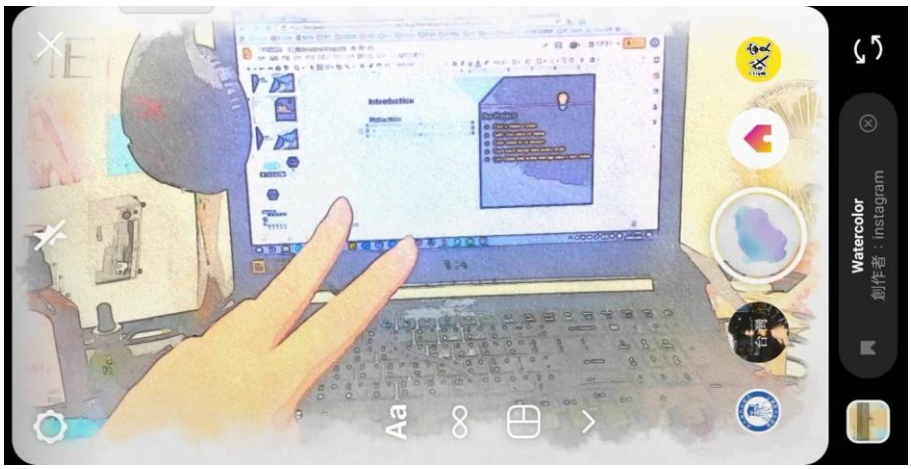
**Group number: 24**
**M11009141 資管碩一 張宜楨、**
**B107909019 四資管四 劉德宇**

## 1. Title:

Making the video into pasta

## 2. Abstract:

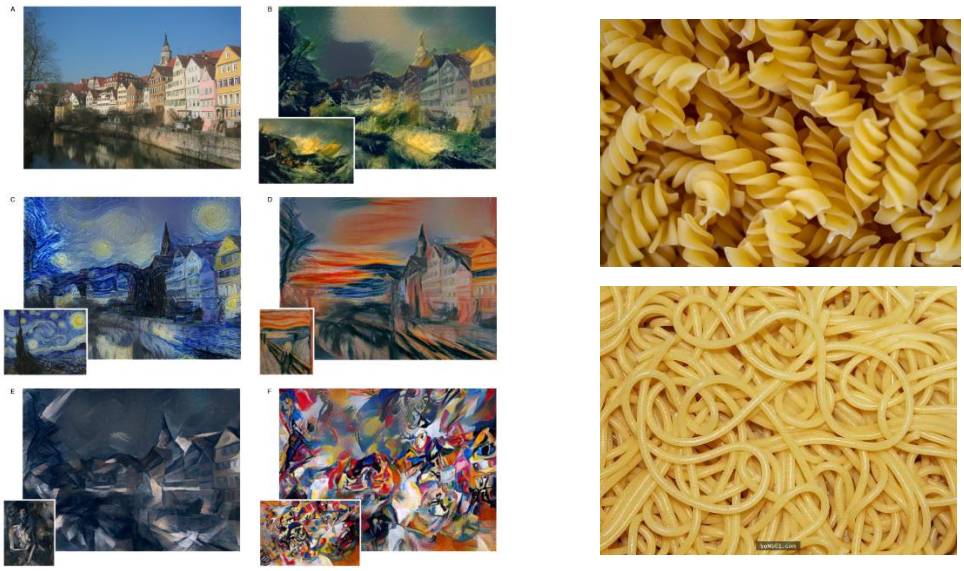
This project is start from a Instagram's filter, this filter can make picture looks like watercolor just in few seconds. But this filter function didn't support to use it on video. So, we decide to make a program not only can change the style, also can be used in any video.




## 3. Introduction:

The program in this project will use a Neural Algorithm to training the model. And then use this model to change each frame in the target video into different style. Because the classical art style like van Gogh's style has been been implemented by a lot of people, so we decide to choose "pasta" style in our project.

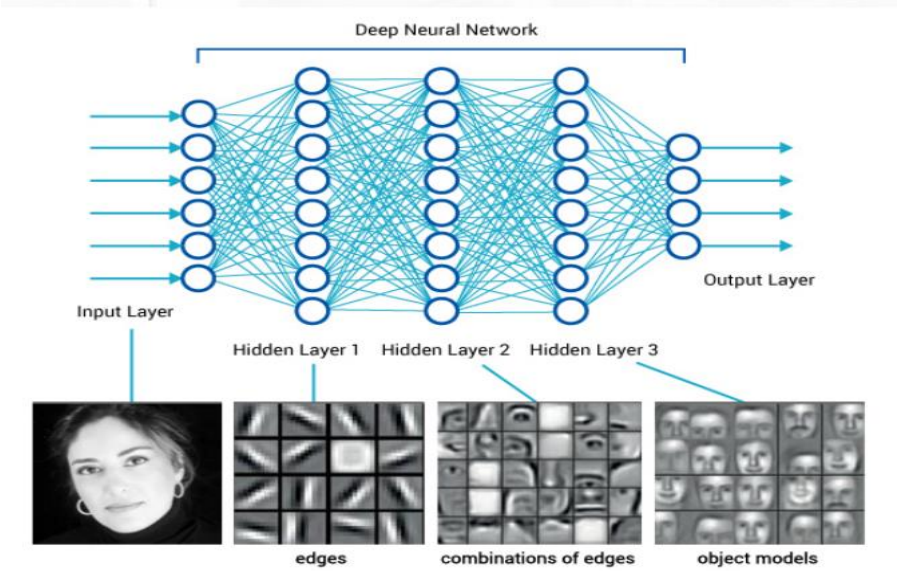
Before starting, make sure your computer has enough memory and GPU, then you can use any video in this program. The style is a part of input in this model, you can change it to any image you want.



the classical art style(left) and the pasta style we used(right)

## 4. Related Work:

**Visual perception**, such as object and face recognition near-human performance, was recently demonstrated by a class of biologically inspired vision models called Deep Neural Networks.



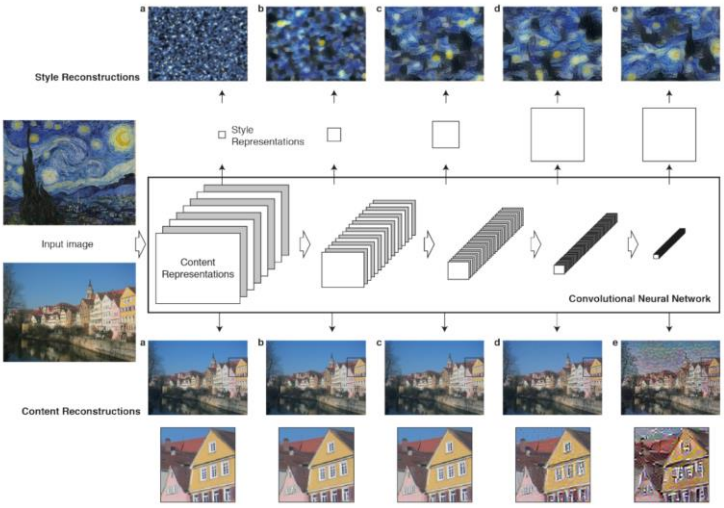

**Deep Neural Networks** mean Imitating the mathematical model of the biological nervous system, multiple calculations, and training to find the optimal and most effective deep learning model.

Here, we used an artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality.

This artificial system used neural representations to separate and recombine the content and style of arbitrary images.

Also, it provides a neural algorithm for the creation of artistic images.

The class of Deep Neural Networks that are most powerful in image processing tasks are called Convolutional Neural Networks. **Convolutional Neural Networks** consist of layers of small computational units that process visual information hierarchically in a feed-forward manner.




When Convolutional Neural Networks are trained on object recognition, they develop a representation of the image that makes object information increasingly explicit along the processing hierarchy. Therefore, along the processing hierarchy of the network, the input image is transformed into representations that increasingly care about the actual content of the image compared to its exact pixel values.

We can directly visualize the information each layer contains about the input image by reconstructing the image only from the feature maps in that layer.

Higher layers in the network capture the high-level content in terms of objects and their arrangement in the input image but do not constrain the exact pixel values of the reconstruction.

In contrast, reconstructions from the lower layers simply reproduce the exact pixel values of the original image.

We refer to the feature responses in higher layers of the network as the content representation.

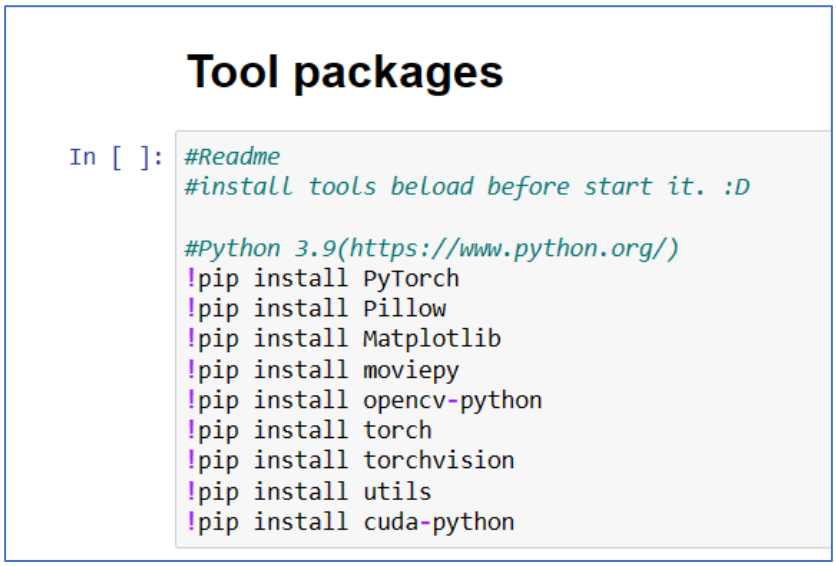
To obtain a representation of the **style of an input image**, we use a feature space initially designed to capture texture information.

This feature space is built on top of the filter responses in each layer of the network. It consists of the correlations between the different filter responses over the spatial extent of the feature maps.

By including the feature correlations of multiple layers, we obtain a stationary, multi-scale representation of the input image, which captures its texture information but not the global arrangement.

## 5. Performance

a、 Set up environment




Using **Python 3.6** or above version, and we should used some tool packages in python

- **PyTorch**
- **Pillow**
- **Matplotlib**
  Use three of tool to computing what wiil image look like with pasta style
- **Moviepy**
  Easier to split or merge audio and video.
- **Opencv-python**
  Extract images from video.
- **Utils**
  A package that stores self-written custom functions.

Make sure your computer has GPU and support **CUDA**, it helps us on calculations other than image processing.

b、 Split the voice at video

提取音檔

```
In [2]: from moviepy.editor import *
        #Jingle Bells- Bongo Cat ver.
        #https://youtu.be/n_8uCh1qLhU
        video = VideoFileClip('test.mp4')
        audio = video.audio
        audio.write_audiofile('test.mp3')

chunk:   6%|▉                                                | 29/454 [00:00<00:01, 235.76it/s, now=None]
MoviePy - Writing audio in test.mp3


MoviePy - Done.
```

c、 Extract images from video

切割影片畫面

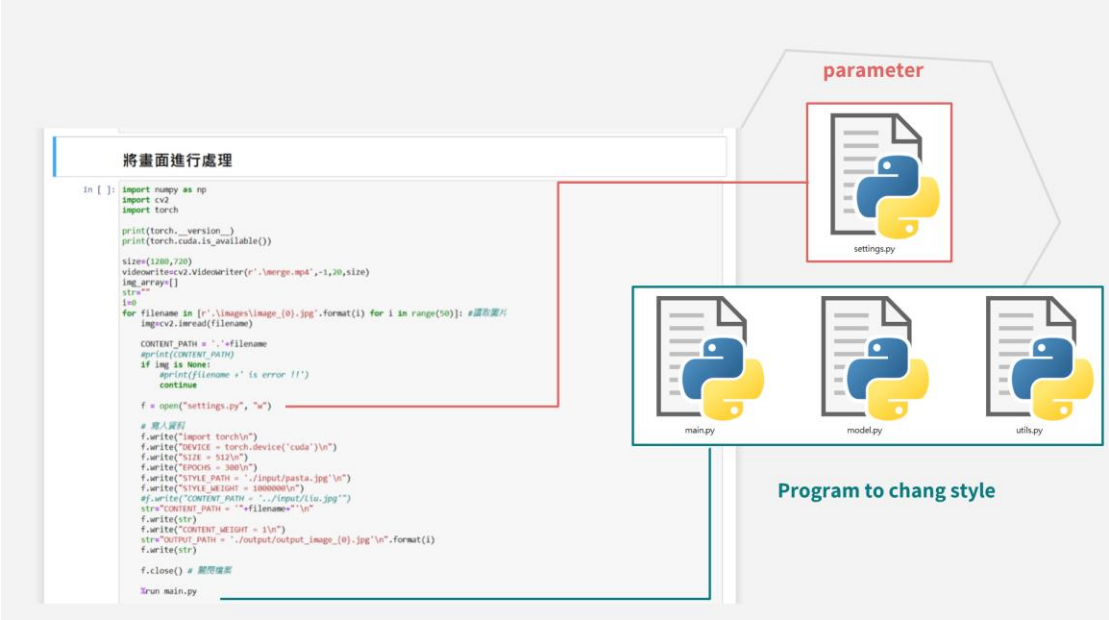
```
In [3]: import os
        import cv2

        filepath = './images/'
        if not os.path.exists(filepath):
            os.makedirs(filepath)

        vidcap = cv2.VideoCapture('test.mp4')
        success, image = vidcap.read()
        fps= int(vidcap.get(cv2.CAP_PROP_FPS))
        count = 0

        while success:
            if count % fps ==0:
                cv2.imwrite(filepath+"image_%d.jpg" % int(count/fps), image)
            print('Process %dth seconds'% int(count/fps),success)
            success, image = vidcap.read()
            count += 1

Process 19th seconds True
Process 19th seconds True
Process 20th seconds True
Process 20th seconds True
Process 20th seconds True
```

The bigger the fps, the longer time you need. And in this simple code we cut it in 1 FPS(frame per second) just want to make it quickly.

d、 Turn each image into pasta style

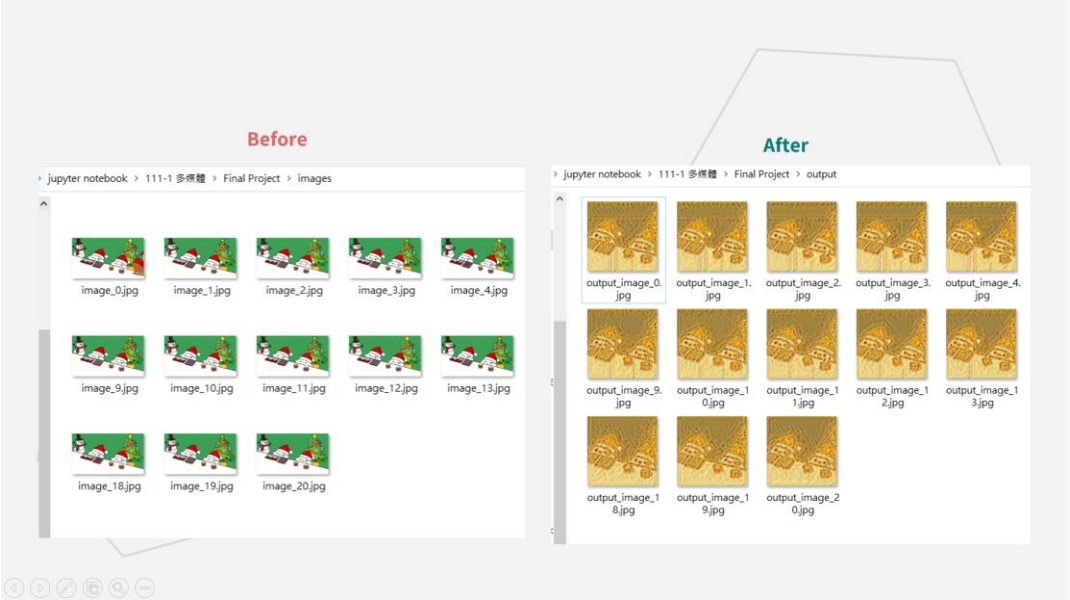
In this step, we throw each image extract before in to model. Let it computing into pasta style.

At the beginning, we try two kind of pasta image as Style input, but it didn't make the output image very clearly to see what it is. To fix this problem, we pick the better output in first round, then use it as style input in second round, and the output become clearly after. Thankfully!




*Setting.py* write our training parameter like DEVICE, EPOCHS, STYLE_PATH, CONTENT_PATH, OUTPUT_PATH, STYLE_WEIGHT.

And *main.py, model.py, util.py* is used to change images style is used to traing images into pasta style. The complete code we put at GitHub (https://github.com/nishikino25/Making-video-into-pasta). I won't go into details here. After it, we got all images with pasta style.




e、 Merage into a new video

Use the output images and origin audio to make new video.
Output video upload on Youtube(https://youtu.be/6RzJxhLA1ys)

```
for filename in [r'.\output\output_image_{0}.jpg'.format(i) for i in range(50)]: #讀取圖片
    img=cv2.imread(filename)

    #print(CONTENT_PATH)
    if img is None:
        #print(filename +' is error !!')
        continue
    img_array.append()


for i in img_array:
    videowrite.write(i)
videowrite.release()
print('merge img end')
```

合併處理後的畫面和音檔

```
video2=VideoFileClip('merge.mp4')
output=video2.set_audio(video.audio)
output.write_videofile("output.mp4", temp_audiofile="temp-audio.m4a", remove_temp=True, codec="libx264", audio_codec="aac")
```

## 6. Conclusion

a、 For the benefit:

Because of Maturity Model, we can change any image as style input. And used any video we want to change.

b、 For the drawbacks

If the video has complex background, its output will be mess.
The program will be very slow when we setting output over 60fps. Mind to figure out is algorithm or hardware limitations make this problem.

## 7. Future work

A future work we think that this program might can use on Universal Design(通用設計), If we can find an image that fits this model, it might can make image's color friendly to colorblind people. Then we can use it on any video or Real time image processing

In order to use it in Real time image processing, we must to consider more related technologies. For example:

**Microservice** → Edge Computing
**5G** → Better transfer speed and IoT communication
**AR Glasses** → The lightest device to use
**Auto focus** or anything I haven't thought of yet.

As the Prof. Yang says, it not only can used change style to help colorblind, it needs more professional point of view to figure out what we actually need, but Universal Design still is an interested direction to do in the future.

## 8. Reference


1. A Neural Algorithm of Artistic Style
   Authors: Leon A. Gatys, Alexander S. Ecker, Matthias Bethge
   https://arxiv.org/pdf/1508.06576.pdf
   tjwhitaker/a-neural-algorithm-of-artistic-style: Neural Style Transfer_ (github.com)
2. Ostagram
   https://www.ostagram.me/
3. Nvidia CUDA tookit
   https://developer.nvidia.com/cuda-toolkit-archive
4. Win10 安裝 CUDA、cuDNN 教學. by 李謦伊 on Medium
   https://medium.com/ching-i/win10-%E5%AE%89%E8%A3%9D-cuda-cudnn-%E6%95%99%E5%AD%B8-c617b3b76deb
5. 一些程式參考
   https://blog.csdn.net/weixin_43202566/article/details/86528845
   https://blog.csdn.net/lr94V587/article/details/127629669
   https://blog.csdn.net/m0_46296922/article/details/109902075