Image Generation from Caption Sentence

목차 Table of Contents

20161616 연제원

20197124 하상욱

20198028 이창현

Part 1
프로젝트 목표

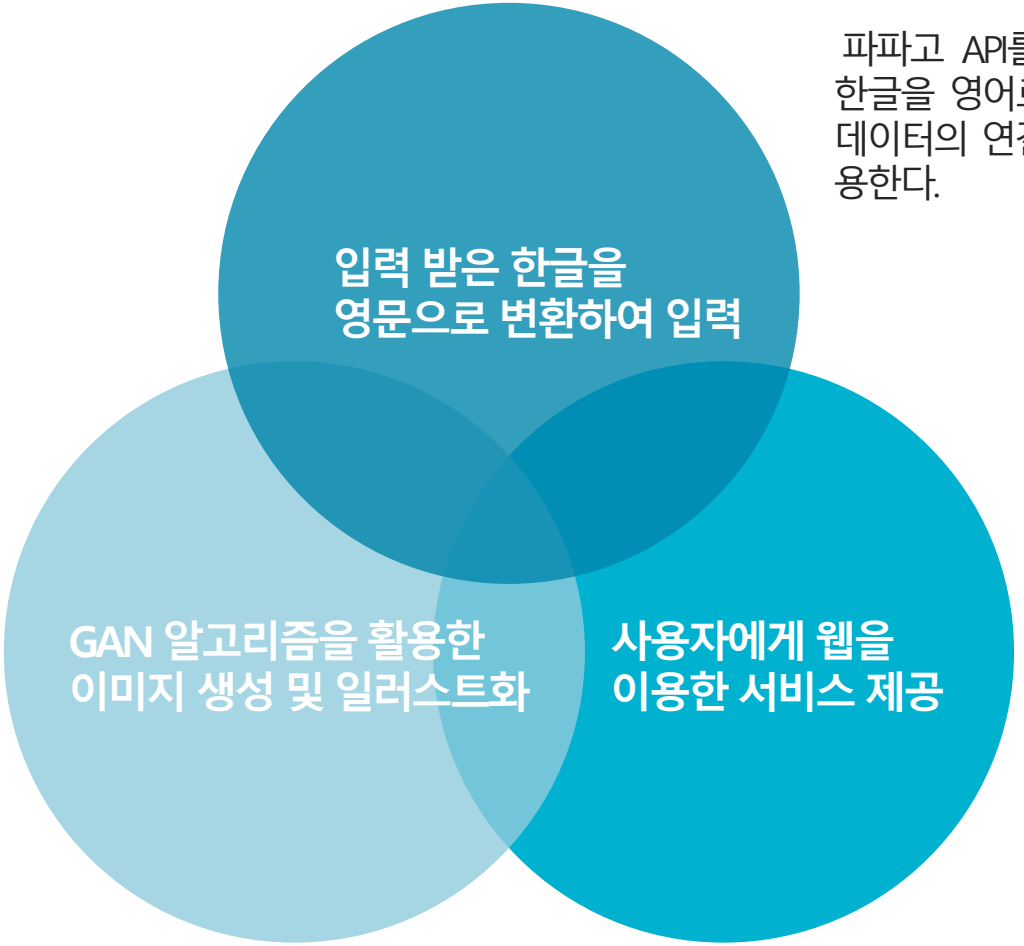
1 프로젝트 목표

대략적인 프로젝트 목표

1	입력 받은 한글을 영문으로 변환하여 입력
2	GAN 알고리즘을 활용한 이미지 생성기능
3	사용자에게 웹을 이용한 서비스 제공

1 프로젝트 목표

프로젝트의 세부적인 목표



파파고 API를 이용하여 입력된 한글을 영어로 번역하여 이미지 데이터의 연결에 CLIP 모델을 활용한다.

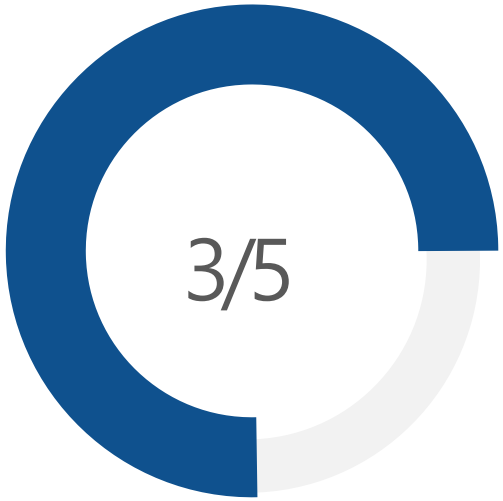
GAN 알고리즘을 이용하여 이미지를 생성한 후 **Sketch-Generation-with-Drawing-Process-Guided-by-Vector-Flow-and-Grayscale** 을 이용하여 일러스트화 한다.

사용자가 웹에서 생성된 이미지를 확인한 후에, 이미지의 여러가지 결과 중 마음에 드는 것과 크기를 선택 가능하게 한다.

Part 2
프로젝트 중간결과

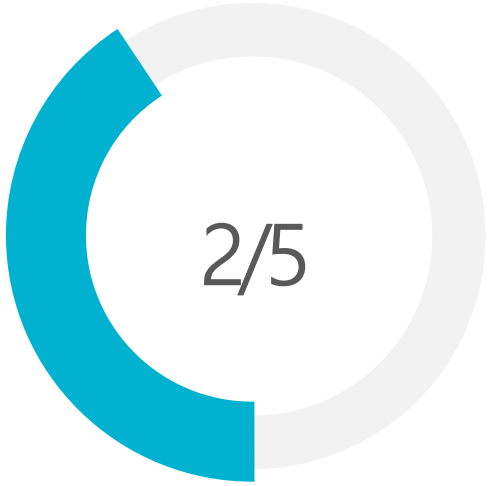
2 프로젝트 중간결과

현재 프로젝트 진행상황



- 한/영 번역 모듈 도입 (O)
- 기초적 예외처리 구현 (O)
- 이미지 데이터와 연결 (O)
- 예외처리 기능 확장 (X)
- Javascript 공격 대응 (X)

자연어 처리



- 필요한 데이터 조건 탐색 (O)
- 학습된 기존 모델 도입 (O)
- 시간적 성능 향상 달성 (X)
- 결결과물의 질적인 향상(X)
- 스타일의 다양성 추가(X)

인공지능 모델

- UI 구성 (X)
- Django 를 이용한 서버구성 (X)

웹 구현

2 프로젝트 중간결과

진행중인 프로젝트 코드

자연어 처리

```python
import requests

def get_translate(text):
    client_id = "--" # <-- client_id 기입
    client_secret = "--" # <-- client_secret 기입

    data = {'text' : text,
            'source' : 'ko',
            'target': 'en'}

    url = "https://openapi.naver.com/v1/papago/n2mt"

    header = {"X-Naver-Client-Id":client_id,
              "X-Naver-Client-Secret":client_secret}

    response = requests.post(url, headers=header, data=data)
    rescode = response.status_code

    if(rescode==200):
        send_data = response.json()
        trans_data = (send_data['message']['result']['translatedText'])
        return trans_data
    else:
        print("Error Code:" , rescode)
```

Base Model Sample

```python
[ ]  # Create the text tokens to feed to the model
     tokens = model.tokenizer.encode(prompt)
     tokens, mask = model.tokenizer.padded_tokens_and_mask(
         tokens, options['text_ctx']
     )

     # Pack the tokens together into model kwargs.
     model_kwargs = dict(
         tokens = th.tensor([tokens] * batch_size, device = device),
         mask = th.tensor([mask] * batch_size, dtype = th.bool, device = device),
     )

     # Setup guidance function for CLIP model.
     cond_fn = clip_model.cond_fn([prompt] * batch_size, guidance_scale)
```

- 본래 Konlpy의 사용을 고려했으나, 기존에 있는 API를 활용하기로 결정

- text-image가 연계된 데이터셋이 해당 기능의 구현에 필요함을 알고 부합하는 방법의 탐색 중 CLIP 모델을 발견

- 한국어 입력을 영어로 변환하는 과정이 모델의 활용에 필요하다고 판단

- 사용 편의성과 익숙함을 고려하여 Papago API를 번역 과정에 활용

- 최종적으로 해당 단계는 현재 한국어 입력 -> Papago API 번역 -> CLIP 으로 이미지 데이터와 연결로 진행됨

2 프로젝트 중간결과

진행중인 프로젝트 코드

이미지 생성

Create Base Model

```
[ ] 1 options = model_and_diffusion_defaults()
    2 options['use_fp16'] = has_cuda
    3 options['timestep_respacing'] = '100' # use 100 diffusion steps for fast sampling
    4 model, diffusion = create_model_and_diffusion(**options)
    5 model.eval()
    6
    7 if has_cuda:
    8     model.convert_to_fp16()
    9 model.to(device)
   10 model.load_state_dict(load_checkpoint('base', device))
   11 print('total base parameters', sum(x.numel() for x in model.parameters()))
```

Create Upsampler Model

```
    1 options_up = model_and_diffusion_defaults_upsampler()
    2 options_up['use_fp16'] = has_cuda
    3 options_up['timestep_respacing'] = 'fast27' #use 27 diffusion steps for very fast sampling
    4 model_up, diffusion_up = create_model_and_diffusion(**options_up)
    5 model_up.eval()
    6
    7 if has_cuda:
    8     model_up.convert_to_fp16()
    9 model_up.to(device)
   10 model_up.load_state_dict(load_checkpoint('upsample', device))
   11 print('total upsampler parameters', sum(x.numel() for x in model_up.parameters()))
```

Create CLIP Model

```
[ ] 1 clip_model = create_clip_model(device=device)
    2 clip_model.image_encoder.load_state_dict(load_checkpoint('clip/image-enc', device))
    3 clip_model.text_encoder.load_state_dict(load_checkpoint('clip/text-enc', device))
```

```
[ ] 1 def show_images(batch: th.Tensor):
    2         """ Display a batch of images inLine. """
    3         scaled = ((batch + 1)*127.5).round().clamp(0,255).to(th.uint8).cpu()
    4         reshaped = scaled.permute(2, 0, 3, 1).reshape([batch.shape[2], -1, 3])
    5         display(Image.fromarray(reshaped.numpy()))
```

```
[ ] 1 #Sampleing Parameters
    2 prompt = trans
    3 batch_size = 1
    4 guidance_scale = 3.0
    5
    6 # Tune this parameter to control the sharpness of 256x256 images
    7 # A value of 1.0 is sharper, but sometimes result in grainy artifatcs.
    8
    9 upsample_temp = 0.997
```

- OpenAI의 glide-text2im에 있는 프로그램을 활용해서 해당 기능을 구현

- Glide 모델이 프로그램에 사용, 텍스트 조건부 이미지 생성이 기능으로 본 프로젝트의 목적에 부합

- 자연어 처리 부분에서 들어온 입력을 토큰으로 받아 이미지 생성에 사용

2 프로젝트 중간결과

진행중인 프로젝트 코드

이미지 생성

Base Model Sample

```
# Create the text tokens to feed to the model
tokens = model.tokenizer.encode(prompt)
tokens, mask = model.tokenizer.padded_tokens_and_mask(
    tokens, options['text_ctx']
)

# Pack the tokens together into model kwargs.
model_kwargs = dict(
    tokens = th.tensor([tokens] * batch_size, device = device),
    mask = th.tensor([mask] * batch_size, dtype = th.bool, device = device),
)

# Setup guidance function for CLIP model.
cond_fn = clip_model.cond_fn([prompt] * batch_size, guidance_scale)

# Sample from the base model.
model.del_cache()
samples = diffusion.p_sample_loop(
    model,
    (batch_size, 3, options["image_size"], options["image_size"]),
    device = device,
    clip_denoised = True,
    progress = True,
    model_kwargs = model_kwargs,
    cond_fn = cond_fn,
)

model.del_cache()

# Show the Output
show_images(samples)
```

* 출처 : https://github.com/openai/glide-text2im

- GLIDE : Guided Language-to-Image Diffusion for Generation and Editing

- 모델 논문 링크
https://arxiv.org/pdf/2112.10741.pdf

2 프로젝트 중간결과

진행중인 프로젝트 코드

생성된 이미지 변환

```
import cv2
import numpy as np
import torch
import torch.nn as nn
import torch.nn.functional as F
import random
import time
import os


from LDR import *
from tone import *
from genStroke_origin import *
from drawpatch import rotate
from tools import *
from ETF.edge_tangent_flow import *
from deblue import deblue
from quicksort import *


# args
input_path = '/content/drive/Shareddrives/CapStone/PaPago_Clip_Sketch/result.png'
output_path = '/content/drive/Shareddrives/CapStone/PaPago_Clip_Sketch/output'

np.random.seed(1)
n =  6                  # Quantization order
period = 4              # line period
direction =  10         # num of dir
Freq = 100              # save every (freq) lines drawn
deepen =  1             # for edge
transTone = False       # for Tone8
kernel_radius = 3       # for ETF
iter_time = 15          # for ETF
background_dir = 45     # for ETF
CLAHE = True
edge_CLAHE = True
draw_new = True
random_order = False
ETF_order = True
process_visible = True
```

- 앞의 이미지 생성만으로는 본 프로젝트의 목적에 완전히 부합하지 못하기에, 스타일 변환 기능의 구현이 필요

- 그림 분위기의 이미지를 만들어 내기 위하여 우선은 Sketch-Generation 모델을 출력된 이미지에 적용

- 현재는 생성된 이미지에 위 모델을 적용한 결과물이 프로그램의 최종 결과

- 해당 모델을 모듈로 직접 도입해 사용, 한 주제를 변환하도록 정해진 부분을 변경해서 프로그램에 구현

* 출처 : https://github.com/TZYSJTU/Sketch-Generation-with-Drawing-Process-Guided-by-Vector-Flow-and-Grayscale

2 프로젝트 중간결과

진행중인 프로젝트 코드

결과

```
[ ]  1 import re
     2 text1 =input('한글을 입력해 주세요. : ')
     3
     4 hangul = re.compile('[^ ㅣ-ㅣ가-힣]+')
     5
     6 trans =  hangul.sub('', text1)
     7
     8 trans
     9
    10 trans = get_translate(trans)
    11
    12 trans
```

```
한글을 입력해 주세요. : 절12벽위^^^의 등3#@#$대
'a lighthouse on a cliff'
```

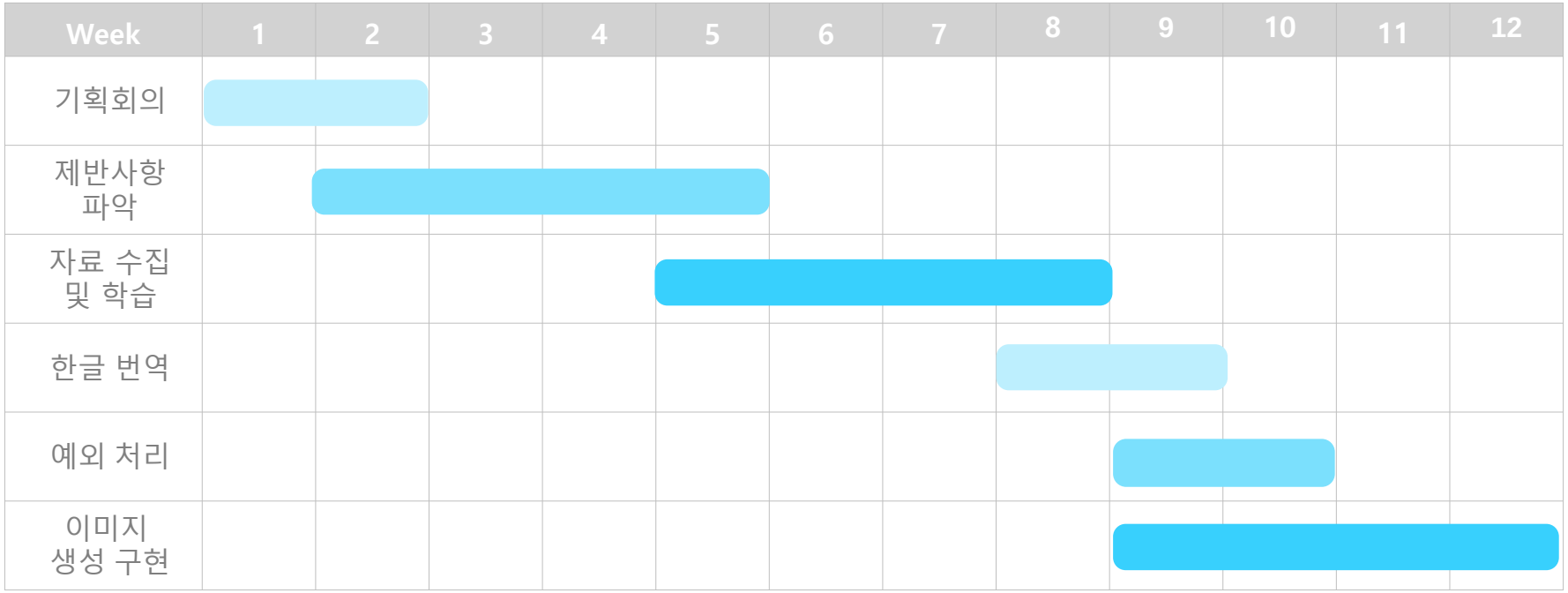
- 지금까지의 실행 결과는 이와 같으며 그림과 같은 이미지를 생성한다는 소기의 목표는 일차적으로 달성된 상태

- 현재로서는 한글을 제외한 특수문자/영어 입력시의 기본적인 예외처리와 같은 기능도 추가됨

Part 3
프로젝트 향후 추진 계획

3 프로젝트 일정표

프로젝트 일정



3

프로젝트 향후 추진 계획

향후 추진계획

Step 1	Step 2	Step 3	Step 4
- 학습데이터 추가 - 결과물 사이즈 변경 - 결과물 스타일 추가	소요 시간 단축	웹사이트 생성 및 적용	유지 보수

>> >> >>

Q n A

THANK YOU