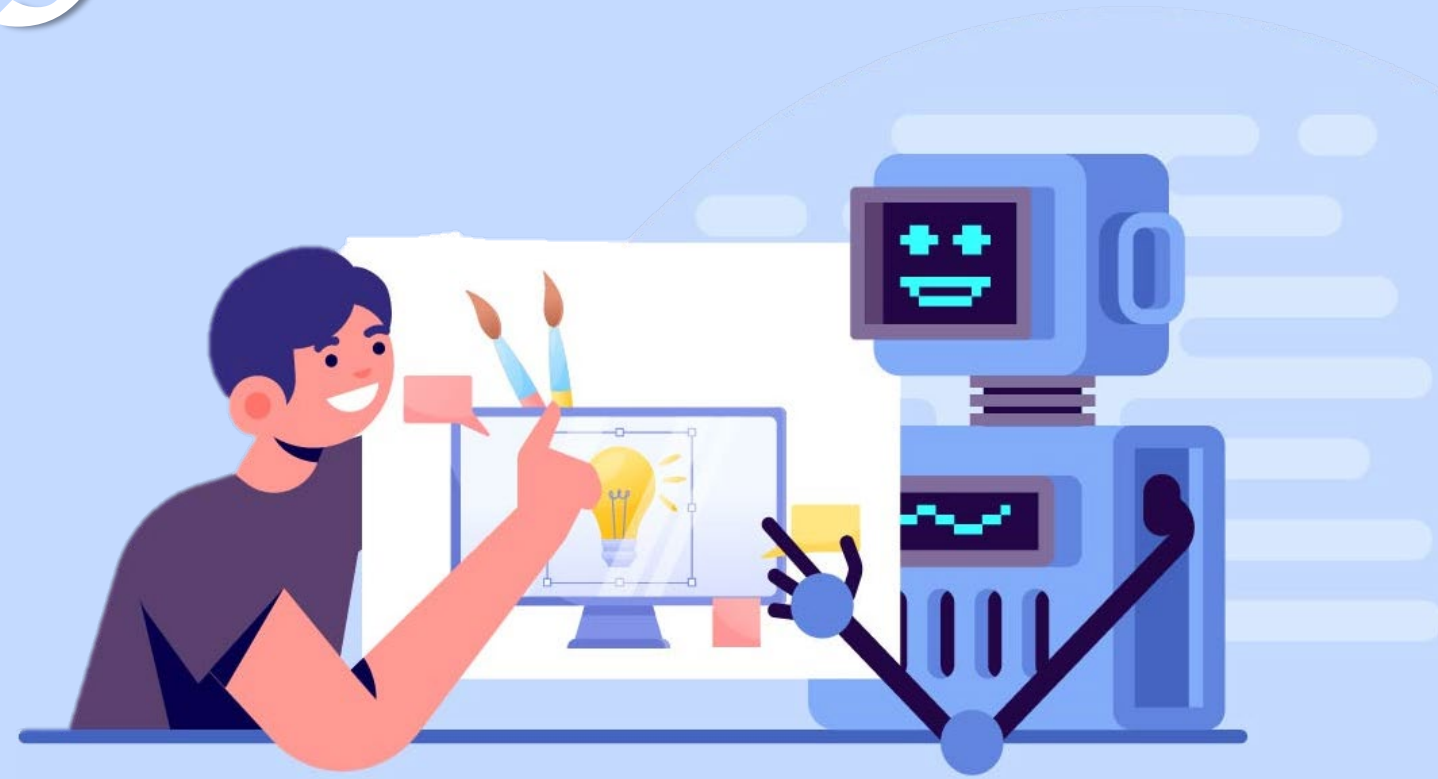
Fancy Font

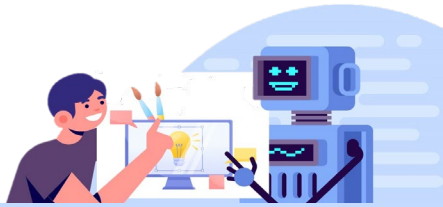


23-2 DSL 모델링 프로젝트 F조

9기 김영현, 유선재, 이성균, 장현빈 & 10기 정성오

Contents

1. INTRODUCTION

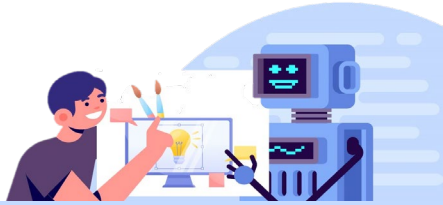


1. INTRODUCTION

Fancy Font

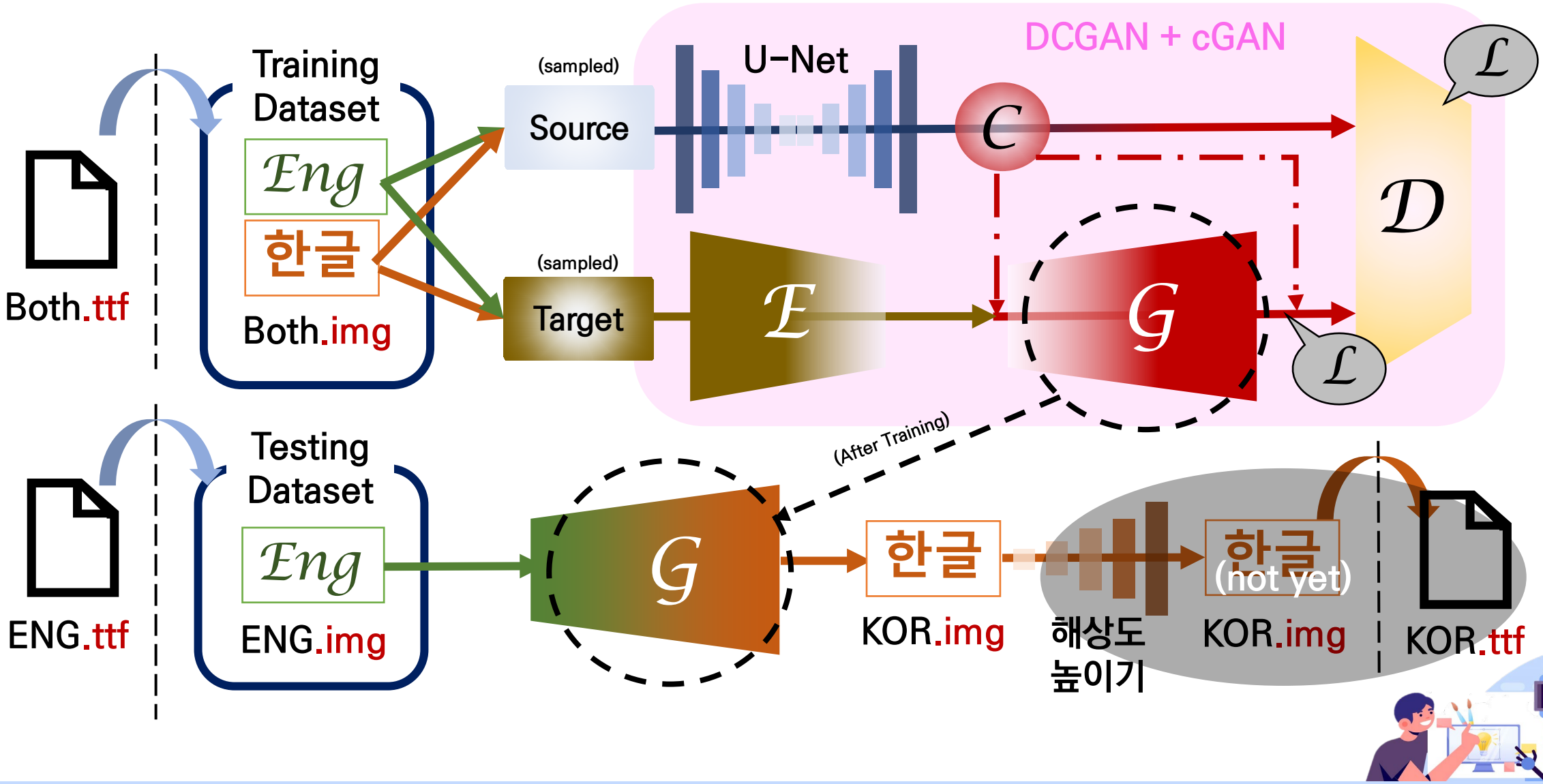
: **한글이 호환되지 않는** 영어 폰트의 한글 폰트를 **생성**하는 모델

Backbone : **DCGAN + conditional GAN**



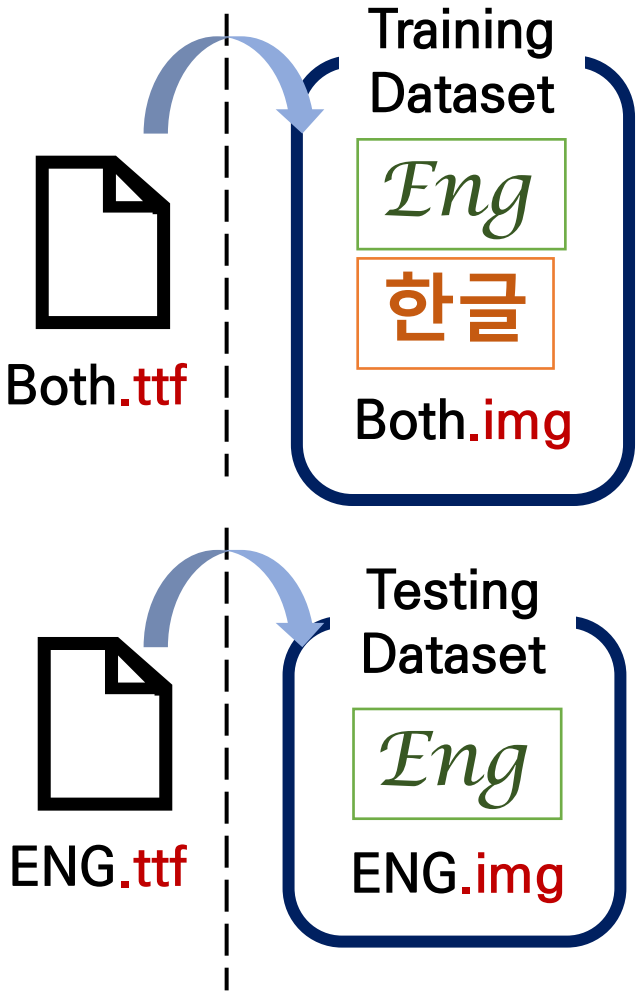
2. OVERVIEW

Fancy Font



3. DATASET

Fancy Font

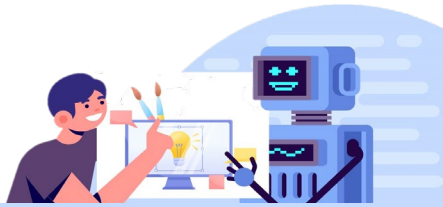


ASCII 코드에 맞춰서 embedding 부여

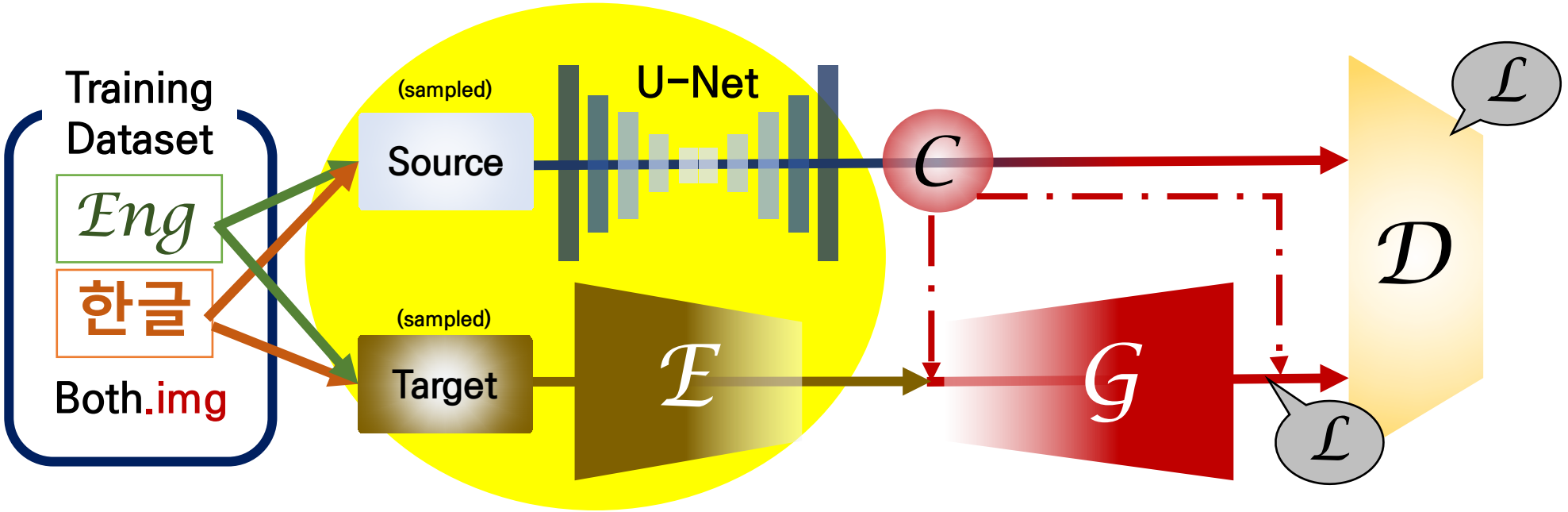
학습시킨 폰트 개수 : 22개(나눔손글씨 우선)

한글 글자 개수 : 11224 글자(초성, 중성, 종성의 조합)

영어 알파벳 개수(대,소문자 구분) : 52 글자

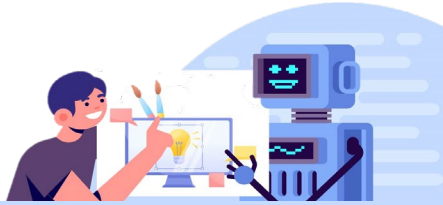


4.1. U-NET

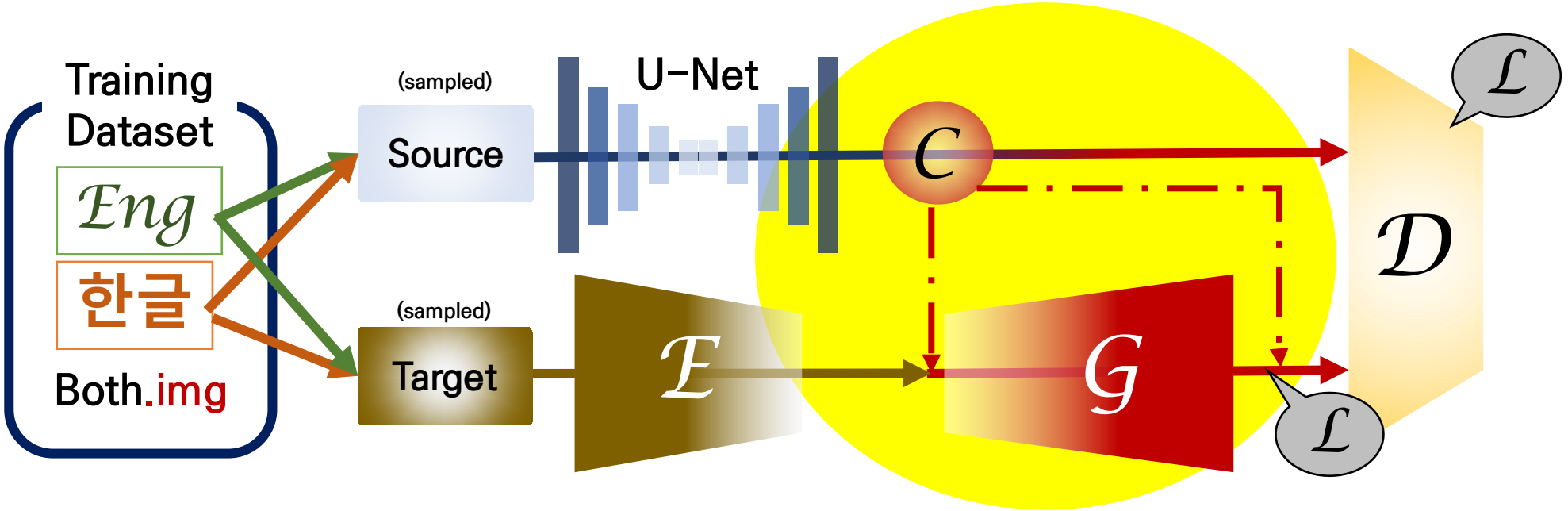


Source font image : 한/영 관계없이 무작위추출
: 폰트의 'structure(글자 구조)'를 뽑아내야 하므로 U-Net 6개 층마다의 input을 condition으로 설정
& U-Net을 통과하여, condition에 structure feature가 잘 들어갔는지 확인

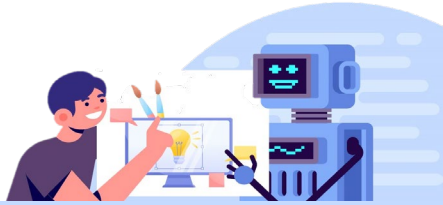
Target font image : 한/영 관계없이 무작위추출
: 'style(글자 모양)'을 뽑아낼 대상이며, Encoder로 style feature를 추출



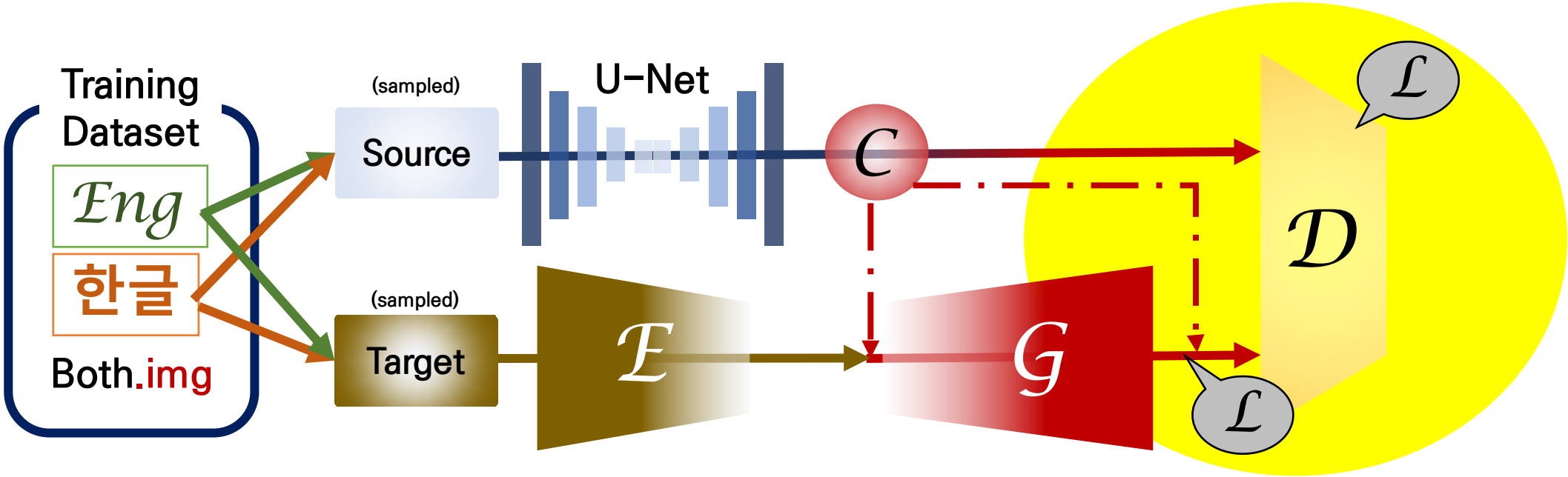
4.2. CONDITIONAL GAN



Encoder : 6단계의 Convolution Block 통과
: 1-32-64-128-256-1024
Generator(Decoder) : 6단계의 DeConvolution Block 통과, 이때 6개 층마다 Source의 condition을 삽입
: 1024-512-256-128-64-32-1
Generator Loss : L1-norm 형태로 계산(절댓값)

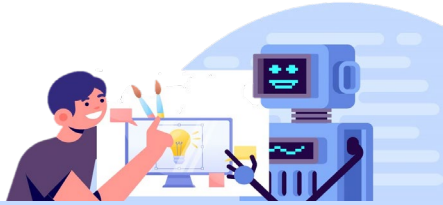


4.3. DISCRIMINATOR



Discriminator : 5단계의 Convolution Block 통과(with max-pooling)
: 2-64-64-64-512-1

Discriminator Loss : 활성화함수(sigmoid)를 통과시킨 후, binary cross entropy 계산
: adversarial loss



4.4. TRAINING

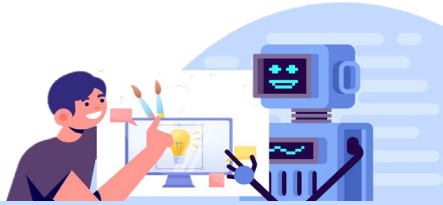
Fancy Font

🎯 Generator Loss : **L1 loss** $\mathcal{L}_{\mathcal{G}}(\boldsymbol{s}, \boldsymbol{s'}) = \sum_{i=1}^{N} |s_i - s'_i|$

🎯 Discriminator Loss : **BCEwithlogits loss** (sigmoid layer + BCEloss)

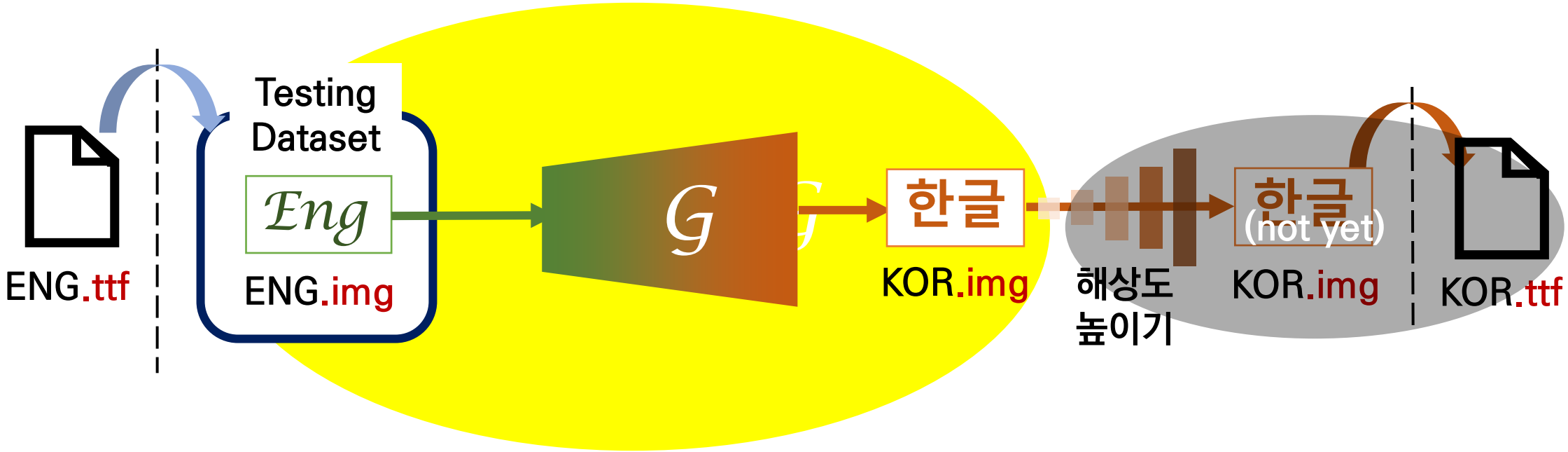
$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{s'}, \boldsymbol{t}, \boldsymbol{w}) = \sum_{i=1}^{N} -w_i \left[t_i \cdot \log \sigma(s'_i) + (1 - t_i) \cdot \log \left(1 - \sigma(s'_i)\right) \right]$$

→ Total Loss = Generator Loss + Discriminator Loss

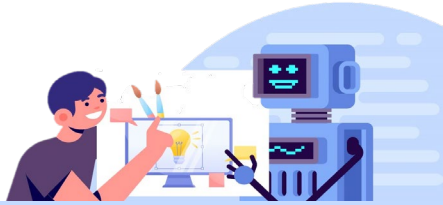


4.5. INFERENCE

Fancy Font

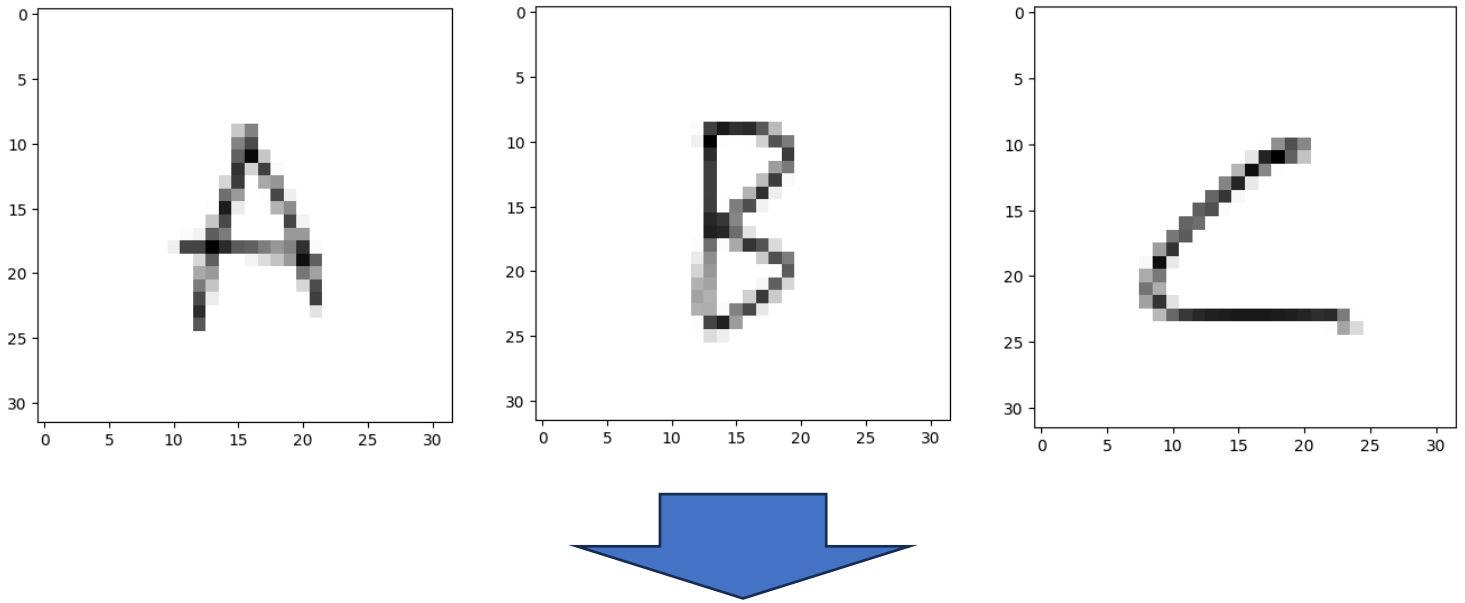


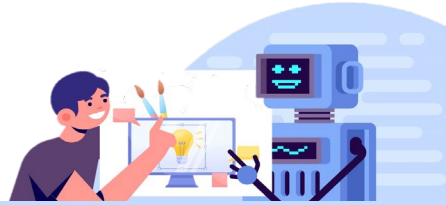
input	output
1) 영어만 지원되는 폰트(의 index) 2) 한글 글자 또는 문장(의 ASCII 코드)	영어 폰트로 쓴 한글 글자의 이미지



5. RESULT

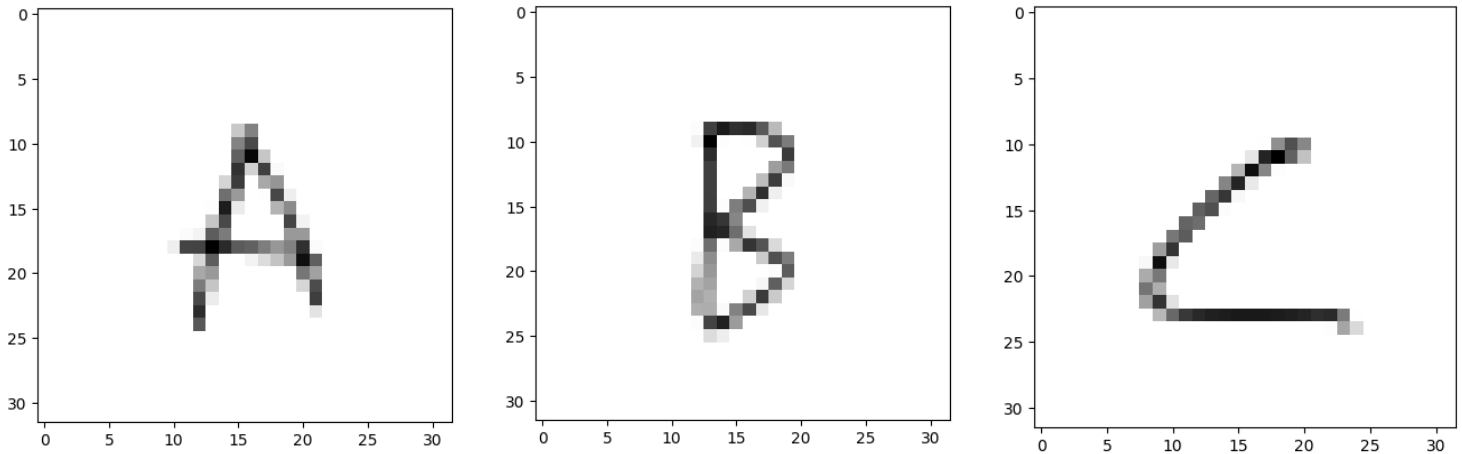
Fancy Font

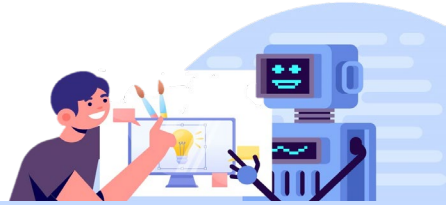




5. RESULT

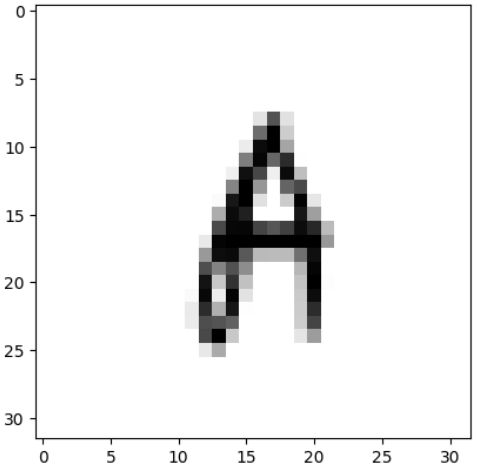
Fancy Font

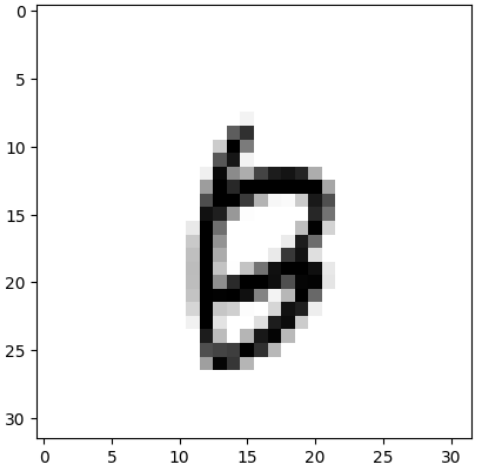


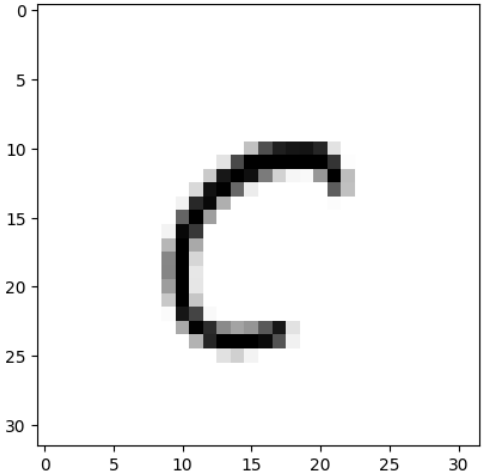


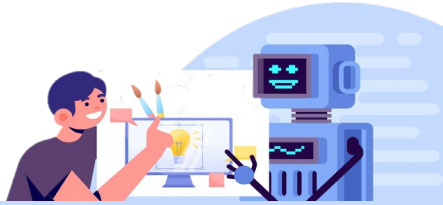
5. RESULT

Fancy Font



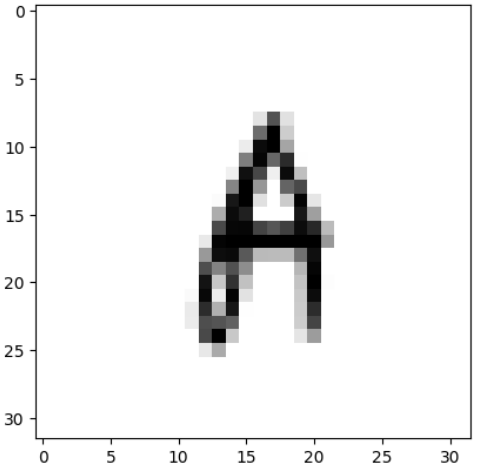


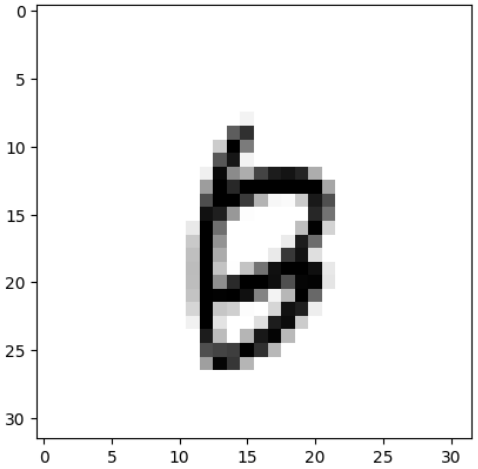


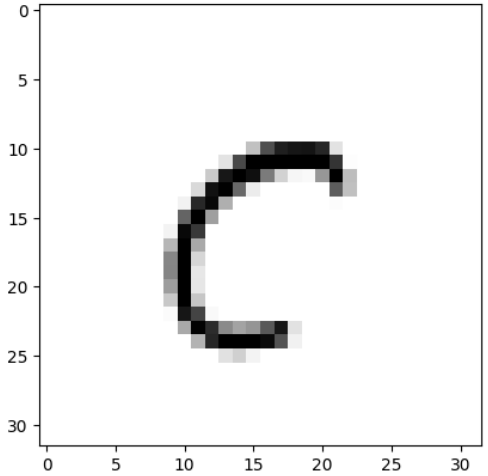


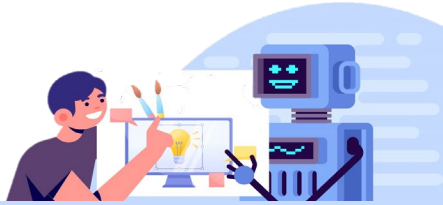
5. RESULT

Fancy Font









6.1. PITSFALL

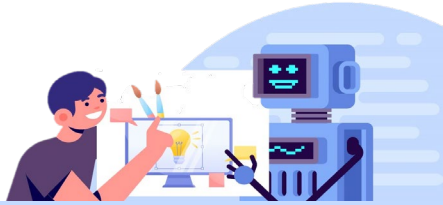
Fancy Font

- 이미지 생성 속도가 다소 느림
  : 한글 글자 조합이 매우 많아서 발생하는 문제(11224개)

- 생성된 이미지의 화소가 낮음(32*32)
  : 모델의 속도를 다소 높이고자 화소를 낮게 처리함

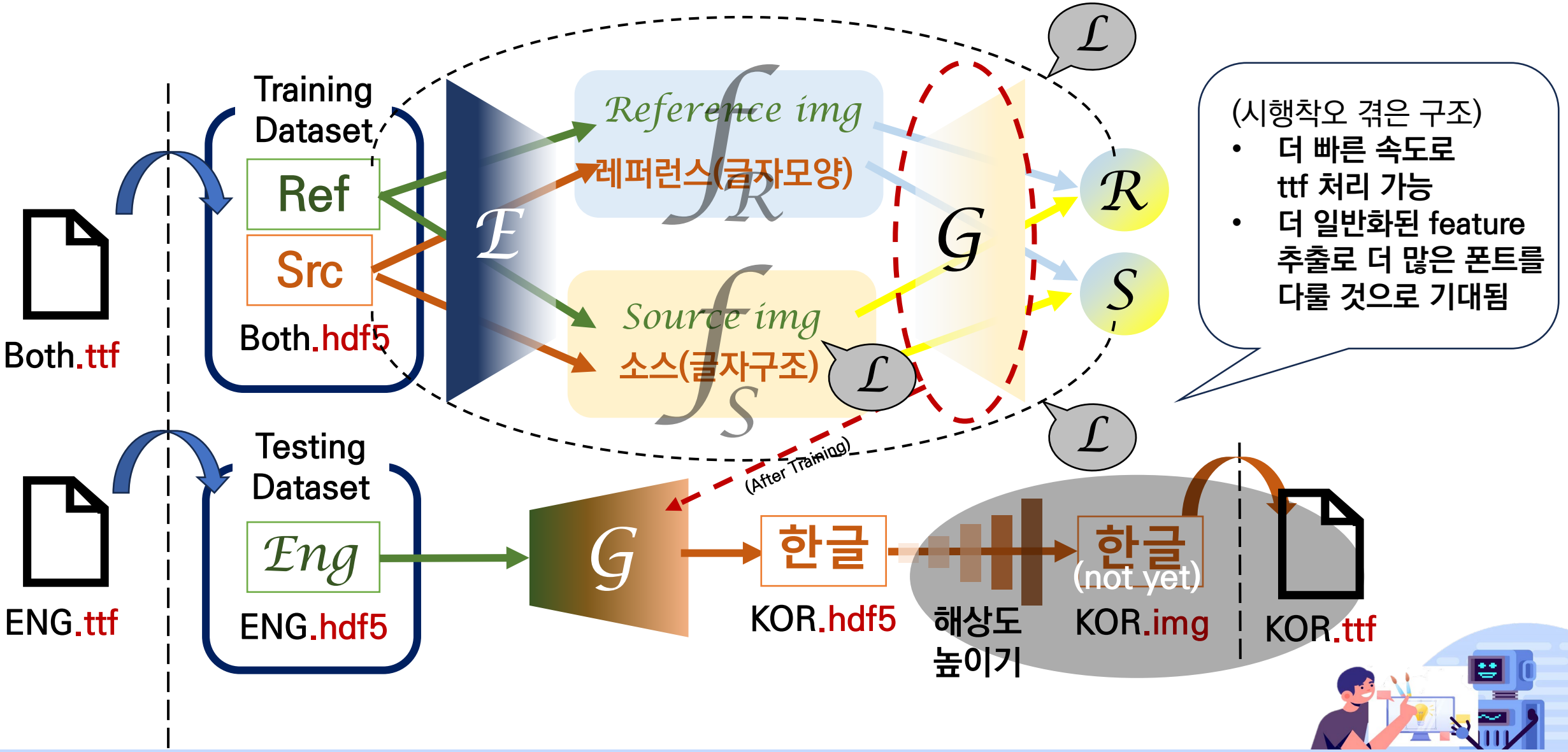
개선방안

a. 데이터셋 처리 속도를 높이기 위해 **메타데이터 형식**으로 데이터셋을 구성하기
b. Source feature(글자 구조에 대한 것) & target feature(글자 모양에 대한 것)을 **깔끔하게 추출하기**



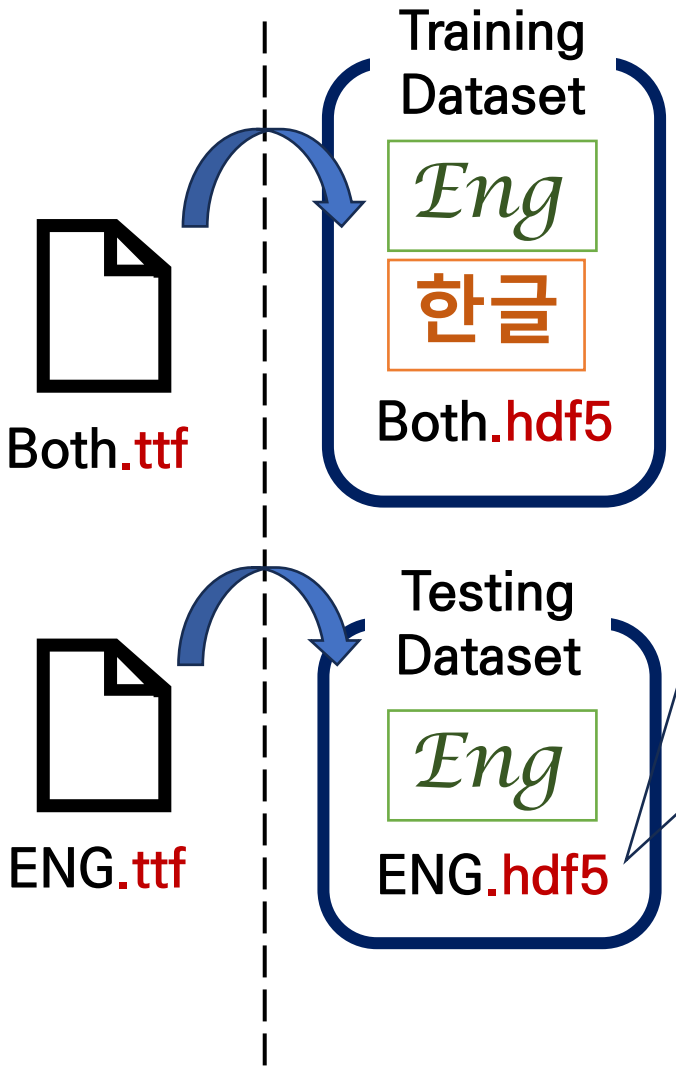
6.2. TRIAL AND ERROR

Fancy Font

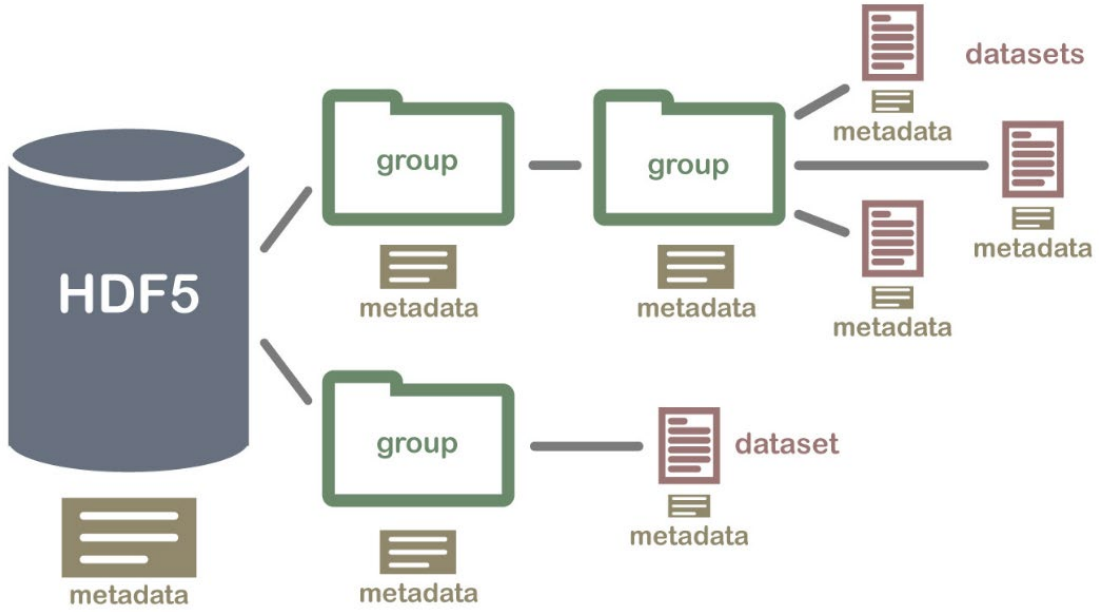


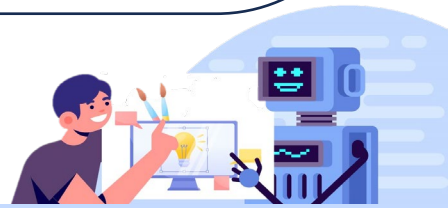
6.3. IMPROVEMENT

Fancy Font



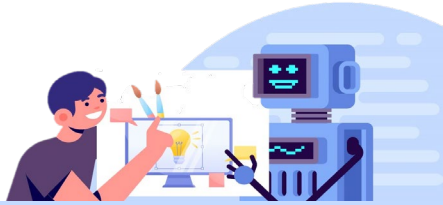
hdf5 (Hierarchical Data Format)
: 대용량 데이터를 빠르게 읽고 쓸 수 있는 **메타데이터** 확장자





6.3. IMPROVEMENT

생성된 이미지(32*32)의 화소를 높여서 선명하게 해주는 부분
: 그래야 궁극적으로 img를 ttf로 바꿀 수 있을 것으로 예상됨
아직 구현하지 못함



Thank you