PORTFOLIO

TABLE OF CONTENTS

ByungJoo Chae

+82 10-28030965|
wpdhkd1642@gmail.com

www.linkedin.com/in/byungjoo-chae-baa661198

Experience

2025.01 –2024.03

DexterStudios
R&D Department
ML Engineer

- 회사에서 필요로 하는 AI 솔루션을 개발하고 배포하는 역할을 하고 있음.

Education

2024.02 – 2022.03

- 충남대 전자공학과 석사과정
- 연구실: CVIP Lab
- 지도 교수: 조동현 교수님

2022.02 – 2016.03

- 충남대학교 전자공학과 학사과정

Project List

Project title	Keywords	Date	Role
Financial Agent AI	Agent AI, LLM	2025.03 -	Developer
Poptale – Translator	LLM, Prompt Engineering	2025.02 -	Developer
De-aging for VFX	Diffusion, De-aging, Video Consistency	2024.05 - 2025.01	Developer
Light Weight Ultra Style transfer Model	LightWeight Model, Ultra Style transfer	2023.12 - 2024.02	Developer
Patch-based Painterly Harmonization for High Resolution Images	Painterly Dataset, Image Harmonization	2023.05 - 2023.12	1st Author (graduation thesis)
Development of Deep Real Technology	Harmonization, Synthetic Dataset	2022.06 - 2023.07	Developer
Online Learning for Reference-Based Super-Resolution	Online learning, Super Resolution, Reference Image	2021.06 - 2022.03	1st Author (MDPI, Electronics)

Projects – Financial Agent AI

Introduction

- 한국투자증권 API를 사용하여 해외 밈 주식 추천과 주식 매매를 할 수 있는 Agent AI 만드는 것을 목표

Duration / People

- 2025.03 ~
- 개발자 4명

Skill

- 기술 스택: langchain, langgraph, Airflow, postgresql
- Language : Python

Contribution

역할 요약 : 개발자 역할 담당

세부 내용 :

- Airflow를 이용하여 주기적으로 해외 사이트 정보를 크롤링하고 postgresql에 저장하는 ETL 파이프라인 구성.
- 크롤링한 정보를 바탕으로 llm을 이용하여 밈 주식 기능을 하는 AI 개발

※

- 해당 프로젝트는 추후 6월 중 오픈소스로 공개 예정.
- 활동 링크 첨부: https://github.com/Pseudo-Lab/Agent_is_all_you_need/discussions/3

Projects – Poptale : Translator

Introduction

- Text-to-image 서비스에서 User 편의성을 위해 한글 입력을 영어 이미지 프롬프트로 바꾸는 것을 목표

Duration / People

- 2025.02 ~
- 개발자 1명

Skill

- 기술 스택: langchain, fastapi
- Language : Python

Contribution

역할 요약 : 개발자 역할 담당

세부 내용 :

- Text-to-image 서비스의 기존 Google 번역기를 사용하던 구조에서 Gemma3 모델을 활용하여 text-to-image 유저의 한글 입력을 영어 프롬프트로 번역하도록 하여 비용 감축함.
- key-value 형태로 입력된 정보를 기반으로 이미지 프롬프트를 생성하는 프롬프트 엔지니어링을 수행하여 자체 서비스에 도입함.

Pipeline

입력: 성별: 여자, 나이 대: 5-year-old, 헤어 색: 딸기우유 색깔, 헤어스타일: 양 갈래로 묶은 머리, 눈 색상: bright hazel, 추가묘사: 항상 웃고 다니며 고양이 인형을 들고 다닌다

출력: A 5-year-old girl has strawberry milk-colored hair styled in pigtails and bright hazel eyes, she is a person who always smiles and carries a cat doll.

[유저의 선택 사항을 프롬프트로 만들어주는 예시]

입력: 그는 행복한 표정을 지으면서, 산 아래에서 등산할 준비를 하고 있다.

출력: He's smiling as he prepares to hike down the mountain.

[한영 번역 예시]

- 유저가 이미지 생성 전 선택할 수 있는 prompt 제공 후 해당 프롬프트를 영어로 된 이미지 프롬프트로 바꿔주기 위해 few shot prompting 하여 진행.

Projects – De-aging for VFX

Introduction

- Drama와 영화에서 요구하는 나이가 어리게 만드는 De-aging solution을 제공 하는 것을 목표

Duration / People

- 2024.04 ~ 2025.01
- Supervisor 1명, 개발자 1명

Skill

- Framework : Pytorch
- Language : Python

Contribution

역할 요약 : 개발자 역할 담당

세부 내용 :

Version 1

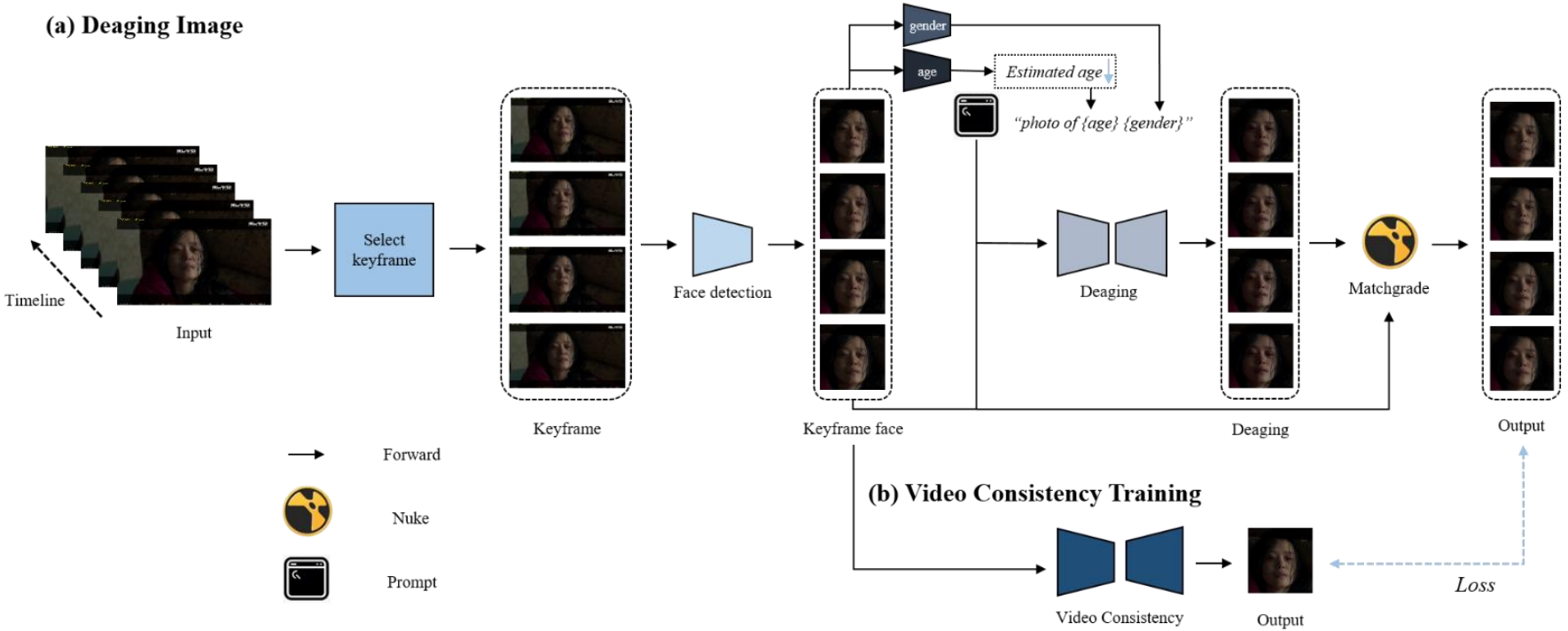
- 디퓨전을 사용한 디에이징 모델을 연구하고, 동아시아 얼굴 데이터셋을 이용하여 모델을 finetuning 하여 디에이징 가능한 연령대를 확장함.

- 프레임 전반에 걸쳐 일관되고 효율적인 디에이징 효과를 보장하는 비디오 디에이징 파이프라인을 개발함.

Version 2

- Version 1의 De-aging Image 모델의 속도를 개선하기 위하여 Instructpix2pix모델을 사용하여 약 90% inference 속도를 줄임.

Pipeline



- De-aging diffusion model Fading을 이용하여 만든 De-aging Image pipeline 및 De-aging Image를 video에 적용할 수 있는 Video Consistency pipeline으로 구성

- 기존 Fading 모델의 경우 age estimation이 서양인 dataset으로 학습되어 동양인의 나이를 너무 어리게 예측하는 경우가 많아 de-aging 할 수 있는 폭을 줄이거나 어린아이를 만들어 냄.

Dataset Re-labeling

Dataset Re-labeling

East Asian Dataset

Pretrained
Age estimation

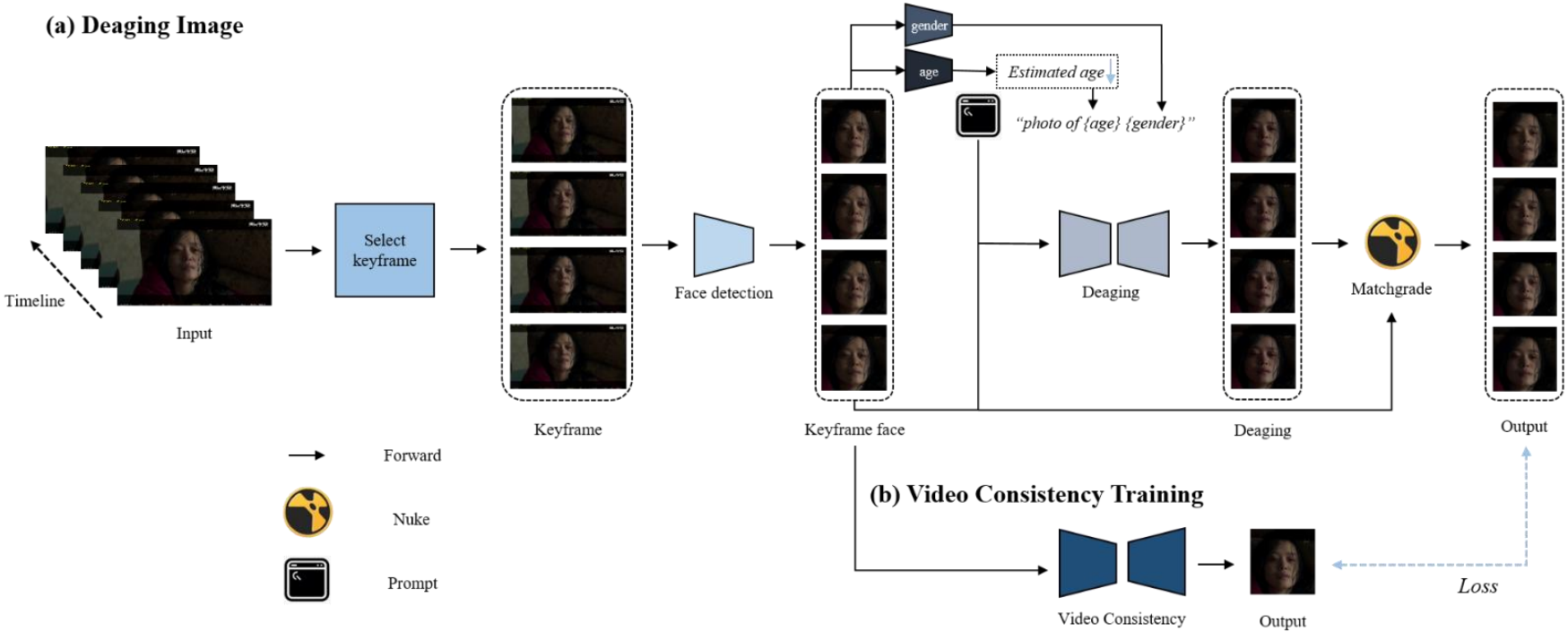
Estimation age + random(10,20)

labeling

- Kaggle에서 East Asian Dataset을 구한 후 pretrained age estimation Dex 모델을 이용하여 labeling 값에 random으로 10~20세를 더하여 labeling 진행

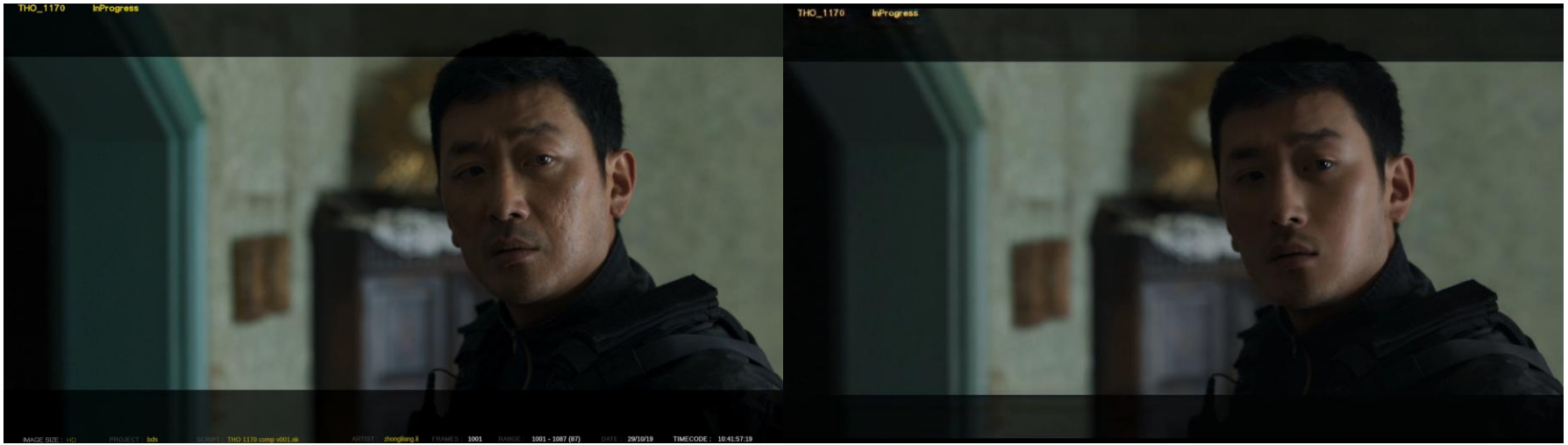
→ 기존 age estimation은 동양인을 10~20세 어리게 예측하는 경향이 있음
(예시: 전지현 기존 age estimation: 20대 초반으로 예측, 실제 나이 : 40대 중반)

Video Consistency



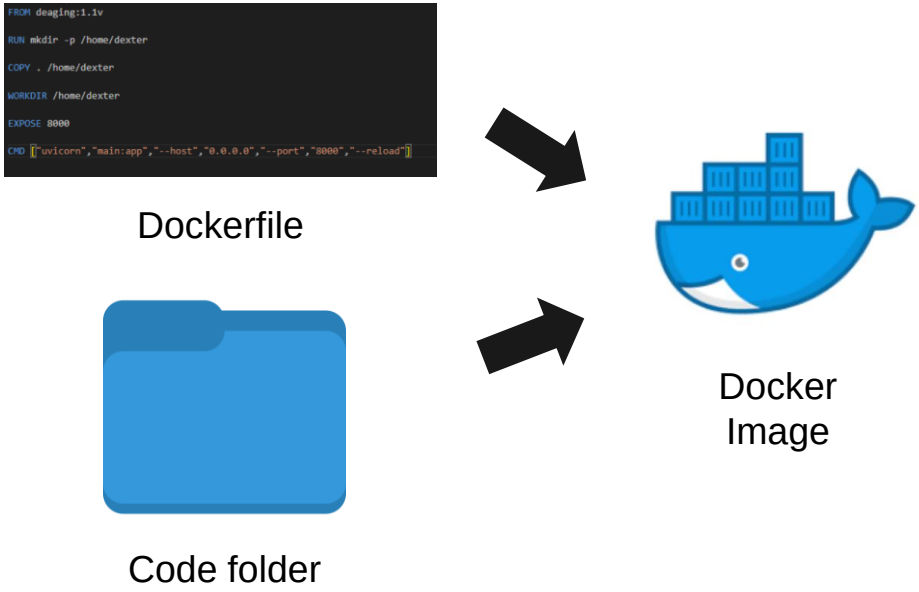
- Video Consistency 모델의 학습은 original image를 입력으로 사용하고, De-aging image를 GT로 사용하여 학습 진행
- 한 인물에 대해서 overfitting을 진행하기 때문에 한 scene의 동일 인물에 대해서는 de-aging된 결과를 만들 수 있음.

Result



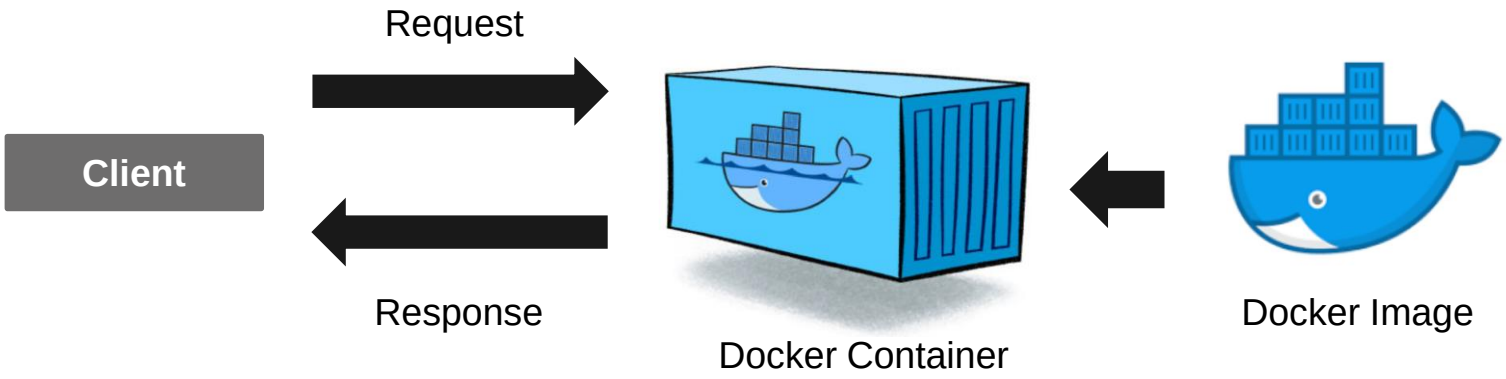
- De-aging 결과 피부 측면에서 깨끗해진 것을 확인할 수 있음.
- Consistency도 잘 유지된 것을 확인할 수 있음.

Deployment and Serving



- Dockerfile을 이용하여 build 하여 Docker Image를 만들어 De-aging solution을 사용할 수 있는 server로 만듬.
- Docker Image에는 server 환경 구축을 하기 위하여 Redis, Celery와 FastAPI를 사용하였음.

Deployment and Serving

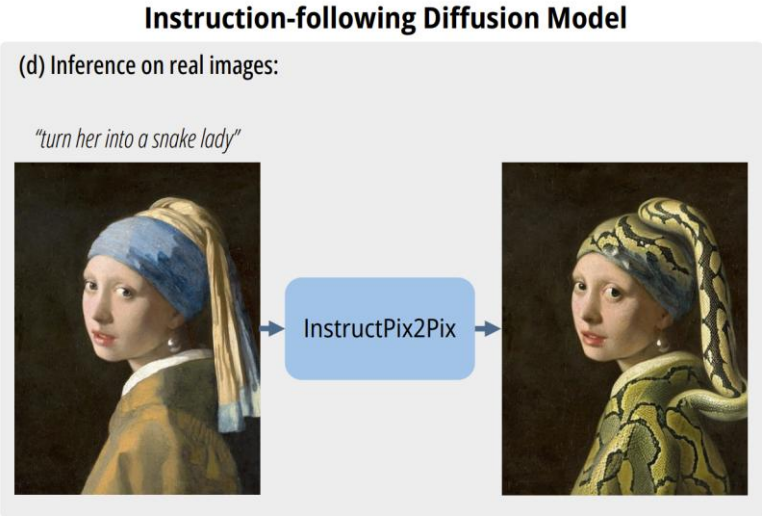


- Client가 httpx를 이용해서 Server에 Request 했을 시에 Server에서 Response 받는 식으로 동작 하도록 구성
- Client는 Nuke 환경에서 실행하도록 하며, Docker image로 만들어 gpu 가 탑재된 각 사용자의 local 데스크탑에 배포하여 사용하도록 함.

→ 이 방법이 server computer를 구축하여 만드는 것 보다 효율적이라고 판단

De-aging Image 0.2v

Purpose : 0.1v의 De-aging Image의 inference 속도가 90sec/image가 걸리기 때문에 속도 개선하는 것을 목표



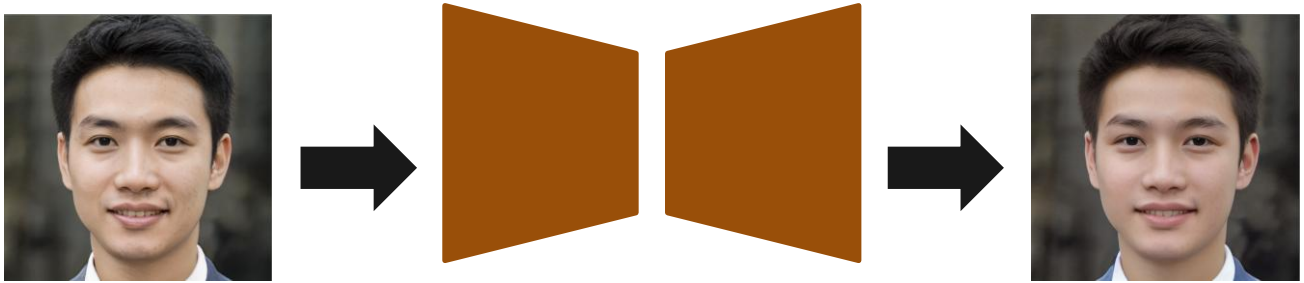
Model

Dataset

- InstructPix2Pix 모델 선택 : prompt-to-prompt 방식이라서 Inference 속도가 빠름.
- Dataset: 동양인 Dataset in kaggle

De-aging Image 0.2v

Data Preparation



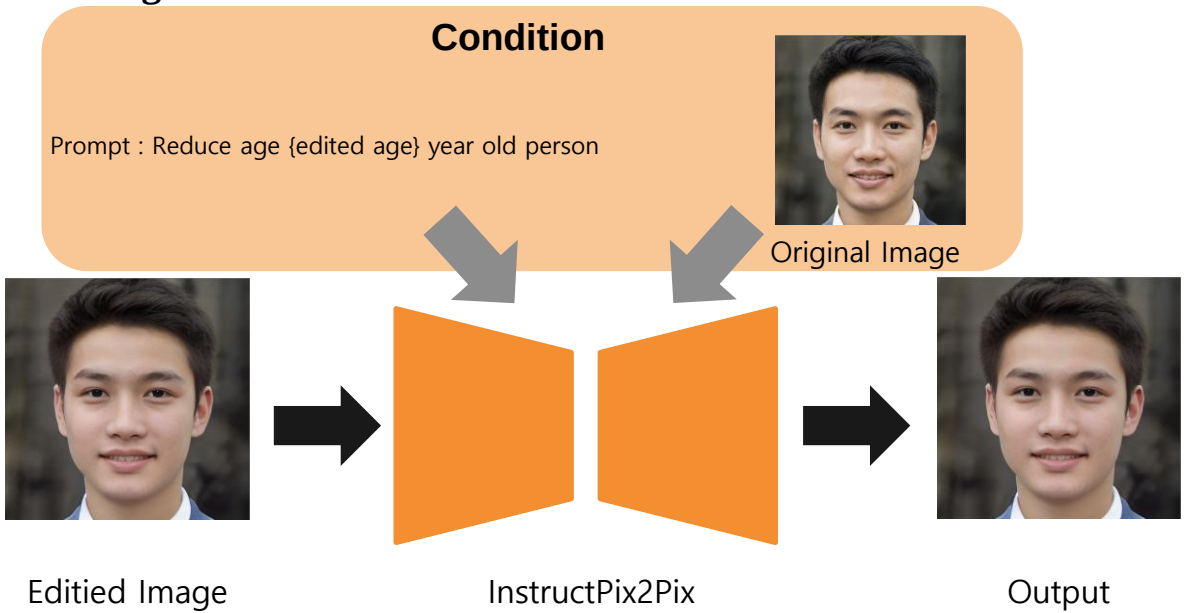
Input Image

De-aging Image

- 기존에 사용하고 있던 De-aging model을 이용하여 Input 이미지를 De-aging 하여 학습에 이용할 Data Pair를 생성
- 기존 나이를 예측 후 50대,40대,30대,20대, 20대 이하에 따라서 각 다른 age 범위를 줘서 Dataset 생성.
- Data 생성 기준은 20세 이상의 사람에 대해서 random으로 De-aging을 함.

De-aging Image 0.2v

Fine-tuning



- 기존에 사용하고 있던 De-aging model을 이용하여 Input 이미지를 De-aging 하여 학습에 이용할 Data Pair를 생성
- Data 생성 기준은 20세 이상의 사람에 대해서 random으로 De-aging을 함.

Projects – Light Weight Ultra Style Transfer Model

Introduction

- Ultra resolution의 Style Transfer를 할 수 있는 light weight model을 개발하는 것을 목표

Duration / People

- 2023.12 ~ 2024.02
- 개발자 1명

Skill

- Framework : Pytorch
- Language : Python

Contribution

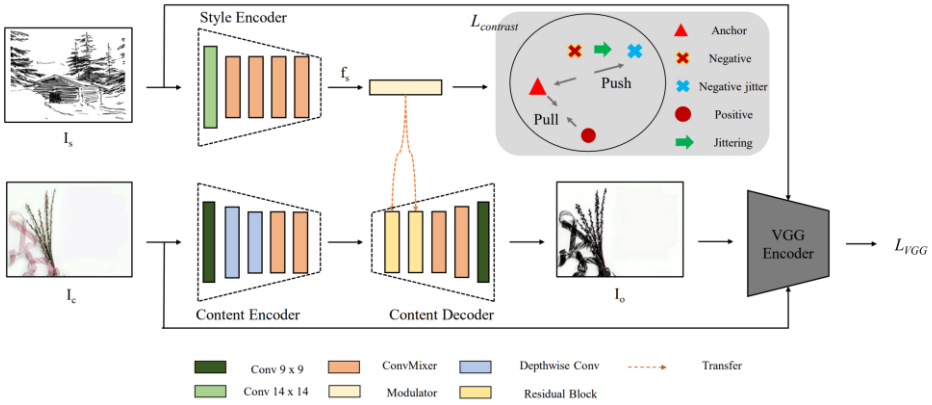
역할 요약 : 개발자 역할 담당

세부 내용 :

- 전체적인 모델의 구조를 ConvMixer라는 모듈을 이용하여 가볍게 만듬.

- Learnable한 module을 이용하여 Global한 정보와 local한 정보를 이용하여 style 정보를 전달

Pipeline



Overall Pipeline

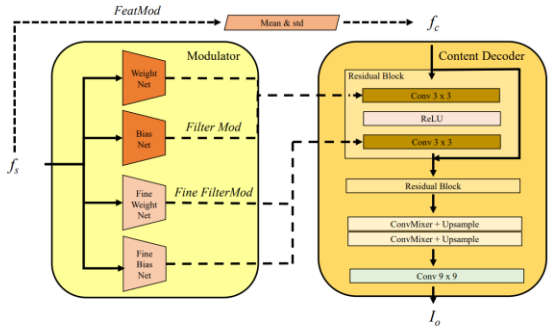
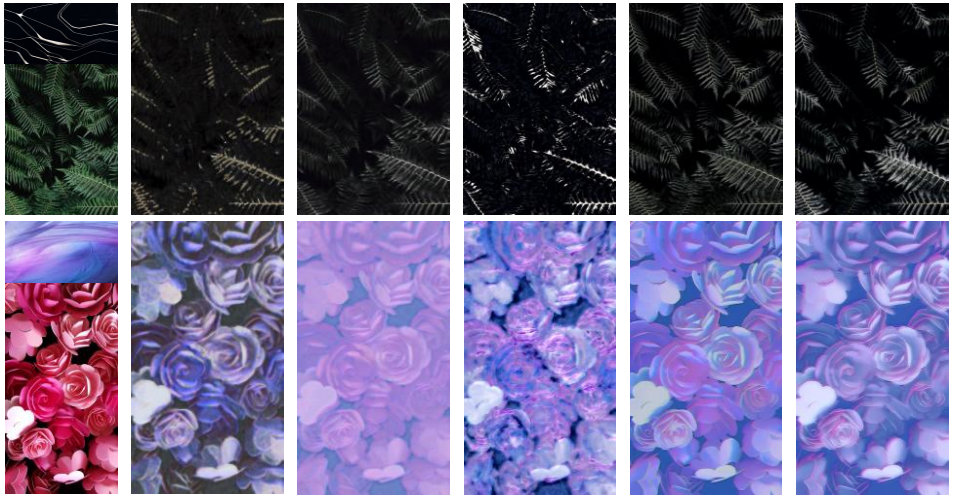


FIGURE 4. **Details of our Triple Modulator**

- ConvMixer 모듈을 이용하여 lightweight backbone을 제작
- Style Encoder로부터 얻게 되는 feature 정보를 증폭시키기 위하여 Triple Modulator 방법을 사용
- Style dataset인 Wikiart dataset의 Class imbalance를 고려하여 Negative sample을 제작할 때 jittering을 사용

Results



Input AdaIN-U LST-U Collaborative -Distill MicroAST Ours

Method	Parmas/10^6	Storage/MB	GFLOPs	Time/sec	Content Loss	Style Loss
AdaIN-U [2]	7.011	94.100	5841.9	3.59	2.086	1.209
LST-U [2]	12.167	48.600	6152.1	3.61	1.920	0.949
Collab-Distill [21]	2.146	9.659	1338.9	8.03	3.363	**0.295**
MicroAST [24]	0.472	1.857	374.9	0.62	2.120	0.584
Ours	**0.306**	**1.239**	**226.2**	**0.56**	**1.990**	0.579

- 가벼웠던 모델인 MicroAST 보다 30% parameter와 GFLOPs를 줄이면서 style 정보 전달과 content 보존을 잘하는 Ultra-Style transfer 모델을 제작

Projects – Patch-based Painterly Harmonization for High Resolution Images

Introduction

- Painterly Harmonization 학습을 위한 Dataset 구축 및 성능 향상을 높이기 위한 것을 목표

Duration / People

- 2023.05 ~ 2023.12
- 개발자 1명

Skill

- Framework : Pytorch
- Language : Python

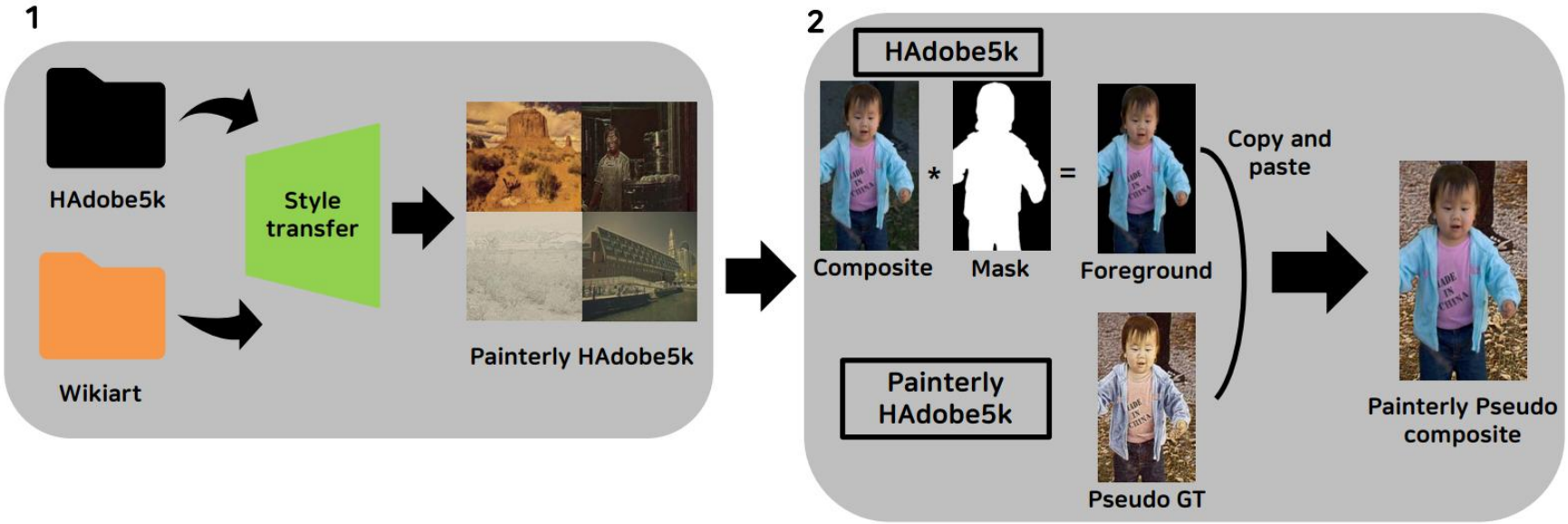
Contribution

역할 요약 : 개발자 역할 담당

세부 내용 :

- Painterly Harmonization Dataset 구축
- Painterly Harmonization 환경에서 patch기반 Image harmonization에서 성능을 높임.

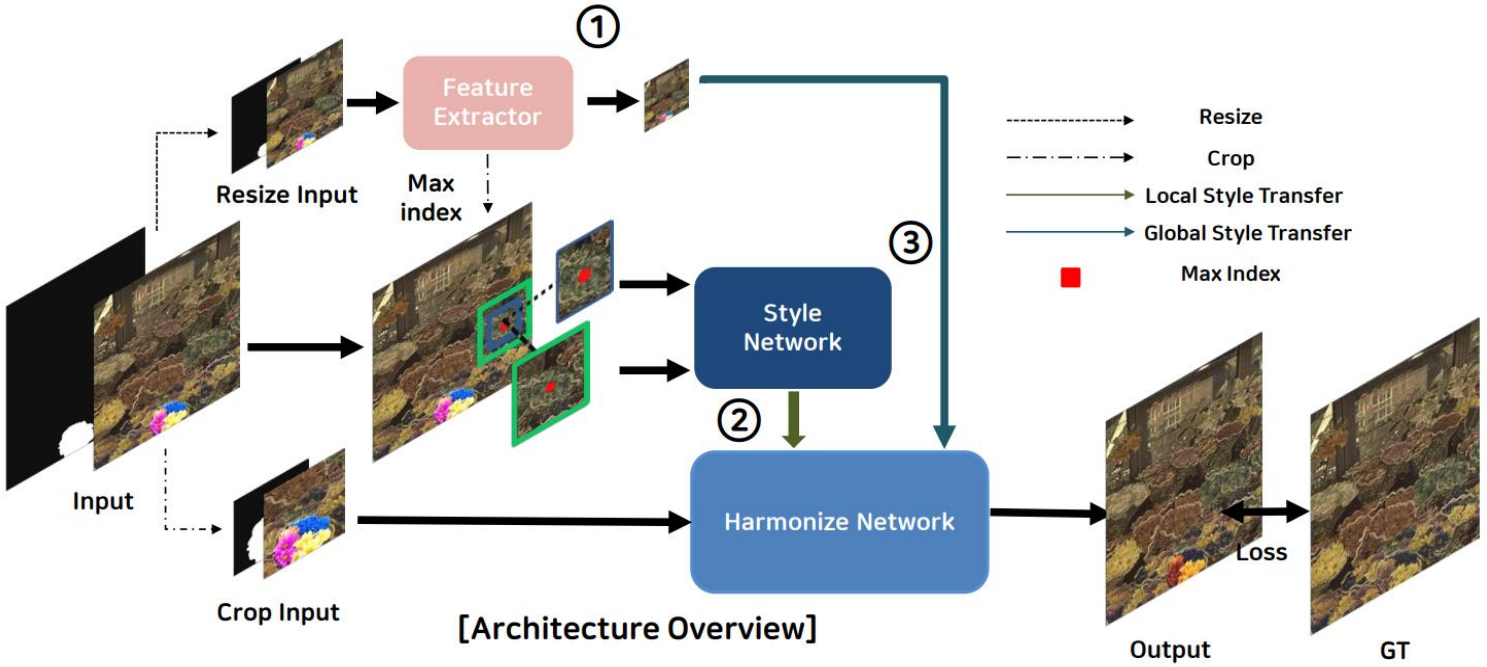
Data Generation



[Data Generate Overview]

- High Resolution Painterly Dataset의 부재로 인하여 Dataset 생성.
- Image harmonization의 benchmark dataset인 HAdobe5k, Style dataset wikiart dataset을 이용하여 생성.

Architecture

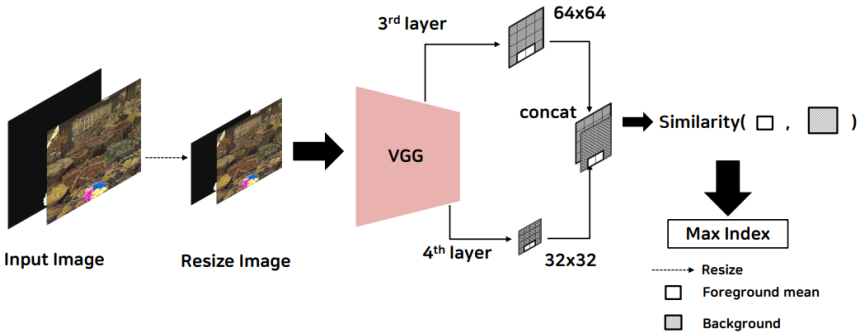


- High Resolution 이미지에 관련된 HRNet-idih를 Base model로 사용.

Model

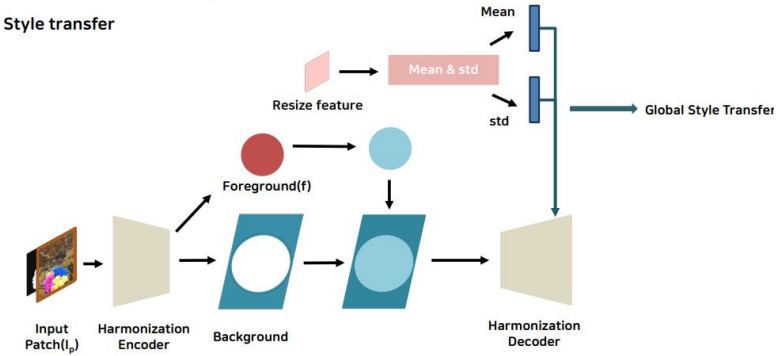
② Patch-wise Method (Feature Extractor)

▶ Style patch selection use VGG Network to use semantic feature



Patch-wise Method(Style Transfer)

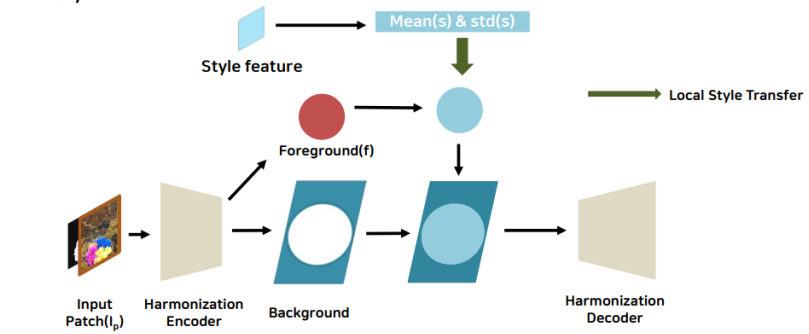
Global Style transfer



Global Style transfer = Decoder_feature * std + mean

② Patch-wise Method(Style Transfer)

Local Style transfer



Local Style transfer = (f – mean(f))/std(f) *std(s)+ mean(s)

- 주변과 유사하게 만들 foreground와 유사한 특징을 갖는 patch를 찾아 style 전달하는 방식으로 진행.
- Global한 정보와 local 한 정보를 모두 전달하기 위한 방식으로 진행.

Results

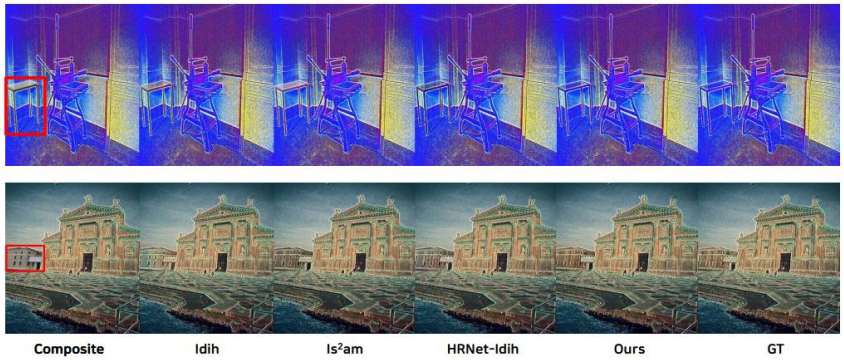
Quantitative Results

1024 x 1024

Model	PSNR	fMSE	MSE
Composite Image	24.12	4360.57	674.27
Idih[5]	28.21	1487.14	252.86
Is^{2}am[5]	30.71	849.09	135.93
HRNET-Idih[5]	30.76	809.16	137.60
Ours	31.11	725.79	127.97

Qualitative Results

1024 x 1024



- High Resolution Painterly Dataset을 제작
- Local한 정보와 Global 정보를 이용하여 성능을 높임.
- Painterly Harmonization 환경에서 patch기반 Image harmonization에서 성능을 높임.

Projects – Development of Deep Real Technology

Introduction

- Synthetic한 환경에서 배경에 전경을 합성할 수 있는 Harmonization에 맞는 Dataset 습득 및 모델 학습하는 것을 목표.

Duration / People

- 2022.07 ~ 2023.06
- 개발자 1명, 업체 1곳

Skill

- Framework : Pytorch
- Language : Python

Contribution

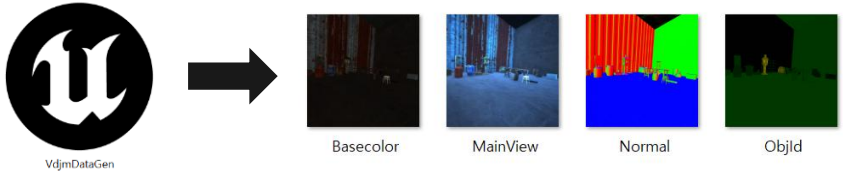
역할 요약 : 개발자 역할 담당

세부 내용 :

- Data Generation 하는 방식을 제안
- Synthetic한 Harmonization dataset 만드는 것을 자동화 함.

Data Generate

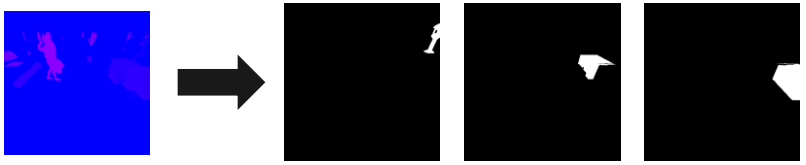
1. Data Generate



Data Generation

Synthetic Dataset

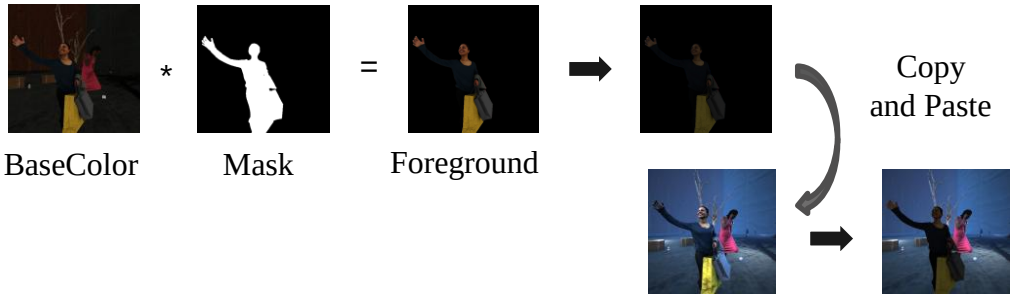
2. Mask Generate



ObjID

Mask 이미지 예시

3. Composite Image Generate



BaseColor

Mask

Foreground

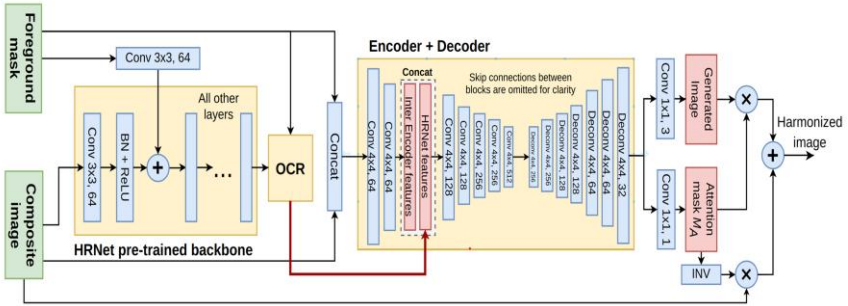
Target
Image

Composite
Image

- Synthetic한 Harmonization dataset 만드는 것을 자동화 함.
- Unreal 환경에서 얻을 수 있는 Raw Dataset으로 부터 학습에 필요한, composite Image, Mask, GT를 얻는 절차를 진행.

Pipeline

Harmonization Model



Image

Input Output GT Mask

Metric

Model	MSE	fMSE	PSNR
Base Model	162.81	2145.53	27.36
Fine-tuning	49.00	549.05	32.43

- 26,157장의 train dataset을 이용하여 Harmonization 모델을 fine-tuning 하고 1,000장의 evaluate dataset으로 평가를 진행함.
- 결과적으로 mask 부분이 주변과 어울리는 결과를 만들 수 있었음.

Projects – Online Learning for Reference-Based Super-Resolution

Introduction

- Reference 이미지를 이용하여 SR 모델의 성능향상을 목표로 함.

Duration

- 2023.05 ~ 2023.12

Skill

- Framework : Pytorch
- Language : Python

Contribution

역할 요약 : 공동 1저자

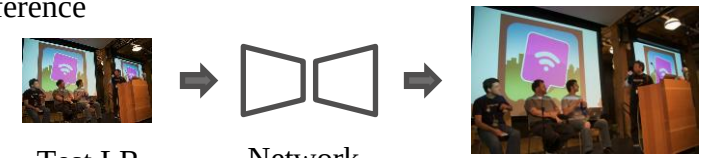
세부 내용 :

- 다양한 데이터 쌍을 학습하기 위한 Reference-based Super Resolution를 위한 온라인 학습 방법 제안
- 간단하지만 효과적이며, SISR와 RefSR 모델 모두에 원활하게 결합될 수 있음
- Reference image와 Input 이미지간의 유사성 정도에 크게 영향을 받지 않고 일관된 성능향상을 보임.

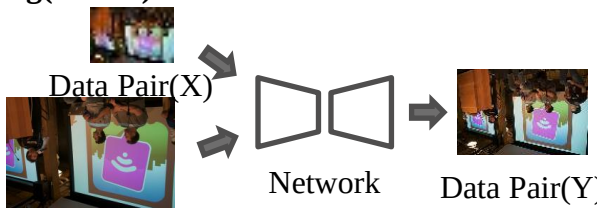
Pipeline

Online learning (SISR)

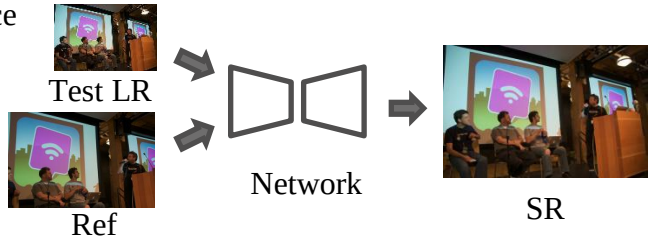
Inference



Online learning(RefSR)



Inference



Model	SISR			RefSR			
Data Pair	D_s^{LR}	D_s^{Pse}	D_s^{Ref}	D_r^{LR}	D_r^{Pse}	D_r^{Ref1}	D_r^{Ref2}
X	$I^{LR}\downarrow$	$\bar{I}^{HR}\downarrow$	$I^{Ref}\downarrow$	$I^{LR}\downarrow$	$\bar{I}^{HR}\downarrow$	$I^{Ref}\downarrow$	$I^{Ref}\downarrow$
R	-	-	-	I^{Ref}	I^{Ref}	I^{LR}	$\bar{I}^{HR}$
Y	I^{LR}	$\bar{I}^{HR}$	I^{Ref}	I^{LR}	$\bar{I}^{HR}$	I^{Ref}	I^{Ref}

- Reference Image를 포함한 다양한 Data Pair에 대해서 실험.
- SISR, RefSR 모두에서 추가적인 모듈없이 좋은 성능을 얻음.

Results

Model	Method	PSNR ↑	SSIM ↑	LPIPS ↓
RCAN	Pre-trained	26.243	0.774	0.2906
Ours + RCAN	D_s^{Ref}	26.703	0.782	0.2856
	$D_s^{LR} + D_s^{Ref}$	26.810	0.785	0.2796
	$D_s^{Pse} + D_s^{Ref}$	26.634	0.781	0.2887
	$D_s^{LR} + D_s^{Pse} + D_s^{Ref}$	26.681	0.782	0.2862

Model	Method	PSNR ↑	SSIM ↑	LPIPS ↓
SRNTT	Pre-trained	25.17	0.734	0.2099
SRNTT-l_2	Pre-trained	26.06	0.765	0.2758
SSEN	Pre-trained	26.412	0.776	0.2915
Ours + SSEN	D_r^{LR}	26.112	0.764	0.2884
	D_r^{Pse}	26.527	0.777	0.2930
	D_r^{Ref1}	26.276	0.770	0.2895
	D_r^{Ref2}	26.675	0.780	0.2874
	$D_r^{LR} + D_r^{Ref1}$	26.257	0.769	0.2946
	$D_r^{Pse} + D_r^{Ref2}$	26.568	0.777	0.2924
TTSR-rec	Pre-trained	27.039	0.799	0.2653
Ours + TTSR-rec	D_r^{LR}	26.812	0.788	0.2545
	D_r^{Pse}	27.337	0.801	0.2663
	D_r^{Ref1}	26.986	0.794	0.2614
	D_r^{Ref2}	27.383	0.802	0.2578
	$D_r^{LR} + D_r^{Ref1}$	26.900	0.790	0.2529
	$D_r^{Pse} + D_r^{Ref2}$	27.400	0.801	0.2663

GT Bicubic SimpleNet EDSR RCAN Ours+ RCAN

SRNTT SRNTT-l2 SSEN TTSR-rec Ours+ TTSR-rec

Results

Model	Kernel	Blind	Method	PSNR	SSIM	LPIPS
Ours+RCAN [7]	$g^d_{0.2}$	-	Pre-trained	17.938	0.497	0.4111
		Non-blind	$D_s^{LR} + D_s^{Ref}$	24.532	0.737	0.2910
	$g^d_{2.0}$	-	Pre-trained	21.131	0.597	0.3335
		Non-blind	$D_s^{LR} + D_s^{Ref}$	26.545	0.783	0.2586
	g^d_{ani}	-	Pre-trained	21.198	0.587	0.3609
		Non-blind	$D_s^{LR} + D_s^{Ref}$	26.414	0.775	0.2679
	$g^b_{1.3}$	-	Pre-trained	25.484	0.738	0.3314
		Non-blind	$D_s^{LR} + D_s^{Ref}$	26.909	0.790	0.2597
	-	-	Pre-trained	21.798	0.606	0.3914
		Blind	$D_s^{LR} + D_s^{Ref}$	24.277	0.692	0.3480

Model	Kernel	Blind	Method	PSNR	SSIM	LPIPS
Ours+TTSR-*rec* [18]	$g^d_{0.2}$	-	Pre-trained	18.415	0.524	0.4039
		Non-blind	D_r^{Ref1}	23.489	0.688	0.3423
			D_r^{Ref2}	24.168	0.717	0.3232
	$g^d_{2.0}$	-	Pre-trained	21.211	0.609	0.3127
		Non-blind	D_r^{Ref1}	25.911	0.760	0.2647
			D_r^{Ref2}	26.561	0.784	0.2624
	g^d_{ani}	-	Pre-trained	21.199	0.596	0.3367
		Non-blind	D_r^{Ref1}	25.512	0.741	0.2841
			D_r^{Ref2}	26.199	0.768	0.2754
	$g^b_{1.3}$	-	Pre-trained	26.147	0.767	0.2912
		Non-blind	D_r^{Ref1}	26.599	0.781	0.2471
			D_r^{Ref2}	26.989	0.796	0.2471
	-	-	Pre-trained	21.820	0.615	0.3603
		Blind	D_r^{Ref1}	23.928	0.672	0.3535
			D_r^{Ref2}	24.010	0.684	0.3461

- Isotropic ,anisotropic한 다양한 kernel에 대해서도 Reference 영상을 사용함으로써 더 좋은 결과를 만들어 냄.

THANKS!

Do you have any questions?

wpdhkd1642@gmail.com
+82 10-2803-0965