



인공지능 체험하기

인공지능 실습하기

숫자를 인식해주는 AI

사람이 마우스로 적는 숫자를 인식하여 그 결과를 알려준다.

(반드시 F5를 누른 전체 화면보기 상태에서 아래 URL을 클릭한다. 전체 해당)

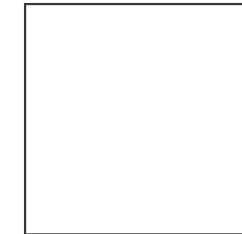
<https://erkaman.github.io/regl-cnn/src/demo.html> (클릭)

[실행 방법] 네모 박스 안에 마우스를 대고 숫자를 적은 후 “Recognize”를 누르면 그 결과를 알려준다. 다른 숫자를 적을 경우 “Clear”를 누른 후 다시 시도한다. 상당히 애매한 경우에도 비교적 정확하게 인식함을 알 수 있다.

GPU Deep Learning Demo

Please draw a digit into this canvas.

(It will probably only work in Firefox and Chrome. And it may not work on mobile. It should look like [this](#))



How does this work?

This demo does handwritten digit recognition by evaluating a Convolutional Neural Network on the GPU with WebGL. The network was trained in TensorFlow [by this script](#), and the network was then reimplemented on the GPU by hand with WebGL. The main purpose of the demo was to demonstrate how our WebGL framework [regl](#) can be used to greatly simplify GPGPU programming in WebGL. The secondary purpose was to test whether evaluating Deep Learning networks in WebGL is doable. To our knowledge, our implementation is the first implementation ever to attempt GPU accelerating neural networks with WebGL. And we hope that this implementation will provide a foundation for people who, like us, wish to experiment with Deep Learning and WebGL. The GPU implementation can be found [here](#).

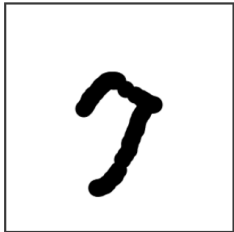
인공지능 실습하기

[실행 결과]

GPU Deep Learning Demo

Please draw a digit into this canvas.

(It will probably only work in Firefox and Chrome. And it may not work on mobile. It should look like [this](#))



7

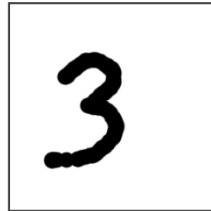
How does this work?

This demo does handwritten digit recognition by evaluating a Convolutional Neural Network on the GPU with WebGL. The network was trained in TensorFlow [by this script](#), and the network was then reimplemented on the GPU by hand with WebGL. The main purpose of the demo was to demonstrate how our WebGL framework [regl](#) can be used to greatly simplify GPGPU programming in WebGL. The secondary purpose was to test whether evaluating Deep Learning networks in WebGL is doable. To our knowledge, our implementation is the first implementation ever to attempt GPU accelerating neural networks with WebGL. And we hope that this implementation will provide a foundation for people who, like us, wish to experiment with Deep Learning and WebGL. The GPU implementation can be found [here](#).

GPU Deep Learning Demo

Please draw a digit into this canvas.

(It will probably only work in Firefox and Chrome. And it may not work on mobile. It should look like [this](#))



3

How does this work?

This demo does handwritten digit recognition by evaluating a Convolutional Neural Network on the GPU with WebGL. The network was trained in TensorFlow [by this script](#), and the network was then reimplemented on the GPU by hand with WebGL. The main purpose of the demo was to demonstrate how our WebGL framework [regl](#) can be used to greatly simplify GPGPU programming in WebGL. The secondary purpose was to test whether evaluating Deep Learning networks in WebGL is doable. To our knowledge, our implementation is the first implementation ever to attempt GPU accelerating neural networks with WebGL. And we hope that this implementation will provide a foundation for people who, like us, wish to experiment with Deep Learning and WebGL. The GPU implementation can be found [here](#).

인공지능 실습하기

주어진 텍스트에 해당하는 이미지를 생성하는 AI

텍스트에 해당하는 이미지를 생성해준다.

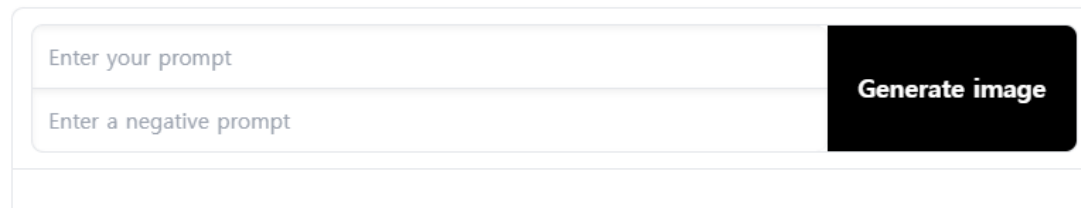
<https://huggingface.co/spaces/stabilityai/stable-diffusion> (클릭)

[실행 방법] 먼저 Example의 3번째 예를 누른 후 “Generate Image”를 체크한다.
(An insect robot preparing a delicious meal)

📦 Stable Diffusion 2.1 Demo

Stable Diffusion 2.1 is the latest text-to-image model from StabilityAI. [Access Stable Diffusion 1 Space here](#)

For faster generation and API access you can try [DreamStudio Beta](#).



The image shows a web interface for the Stable Diffusion 2.1 Demo. It features two input fields: "Enter your prompt" and "Enter a negative prompt". To the right of these fields is a black button with the text "Generate image" in white. Below the input fields, there is a horizontal line, and below that, a vertical line, suggesting a grid or a list of generated images.



인공지능 실습하기

[실행 결과]

그 결과 다음과 같은 “맛있는 음식을 준비하는 로봇” 이미지가 나타난다.

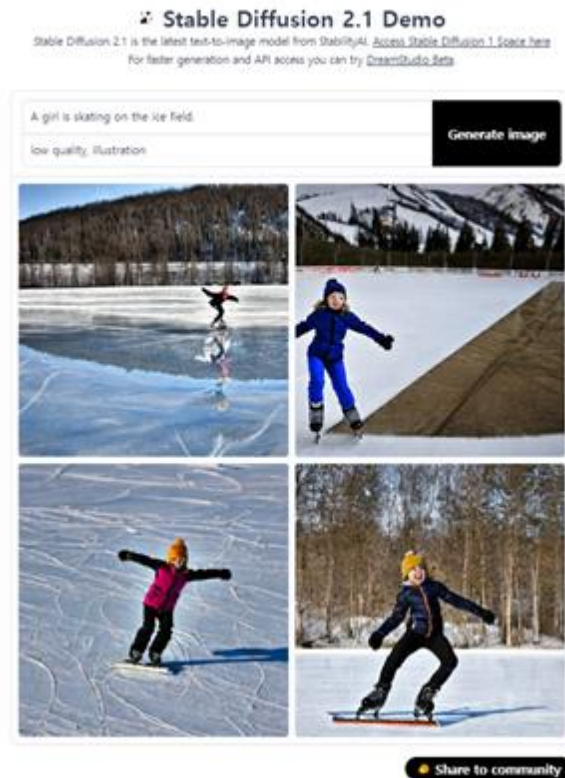


인공지능 실습하기

[실행 결과 2] 그리고 다음과 같은 문장을 Copy하여 상단의 “Enter your prompt”에 넣거나 타이프하고 난 후 “Generate Image”를 체크한다.

A girl is skating on the ice field.

그 결과 다음과 같은 이미지를 얻을 수 있다.



참으로 신기하다. 다음의 2가지 경우도 넣어서 그 결과를 살펴본다.

- A dog and cat is playing in the ground.
- Peoples are dancing on the floor.

인공지능 실습하기

흑백 이미지를 컬러 이미지로 복원해주는 AI

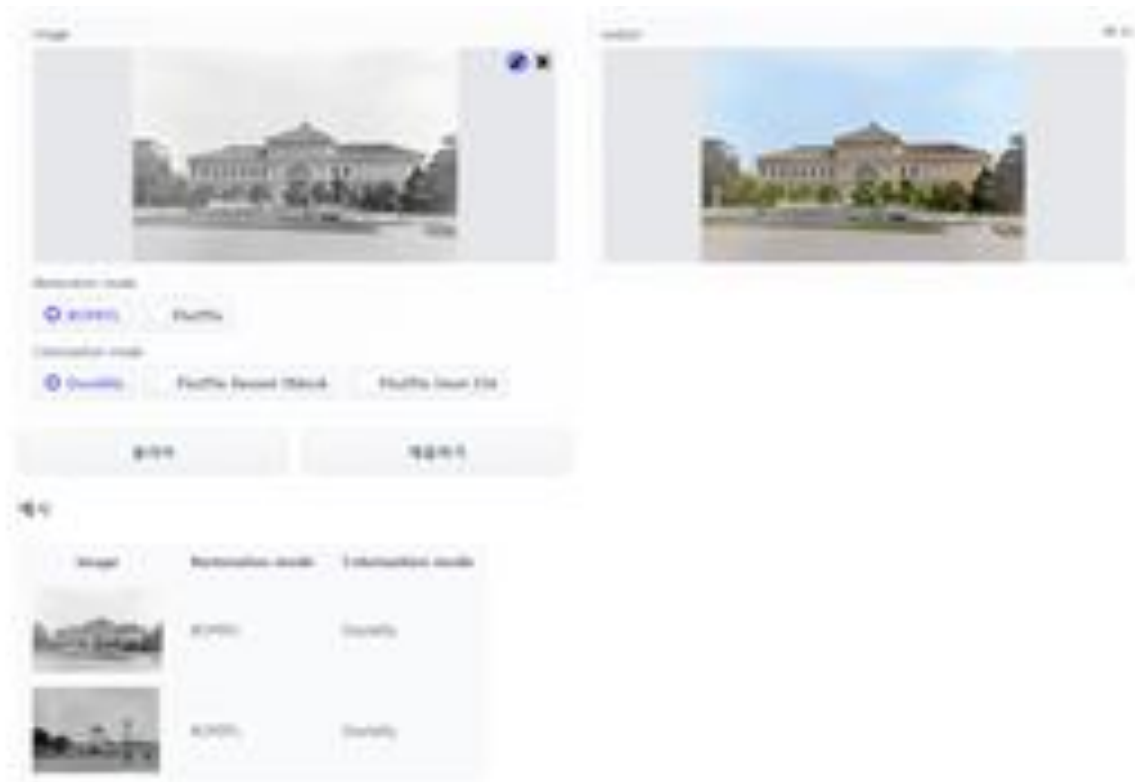
https://huggingface.co/spaces/manhkanhUIT/Image_Restoration_Colorization

[실행 방법] 예제에서 첫 번째 이미지를 선택하여 “제출하기”를 누르면 한참 후 오른쪽에 Color 영상으로 복원되어 나타난다.
그 후 클리어를 누르고 Colorization Mode를 바꾸어가며 변화를 살펴볼 수 있다.
또 여러 가지 이미지를 선택할 수 있다.
물론 “클리어”를 누른 후 점선 박스를 클릭하여
물론 “클리어”를 누른 후 점선 박스를 클릭하여
컴퓨터에 저장된 디렉토리/파일을 찾아 업로드하여
실습할 수도 있다.



인공지능 실습하기

[실행 결과] 오른쪽에 Color 영상으로 복원되어 나타난다.



인공지능 실습하기

이미지에서 실루엣을 스케치해주는 AI

얼굴과 같은 비교적 단순한 그림부터 다소 복잡한 풍경에 이르기까지 실루엣을 스케치하여 보여준다.

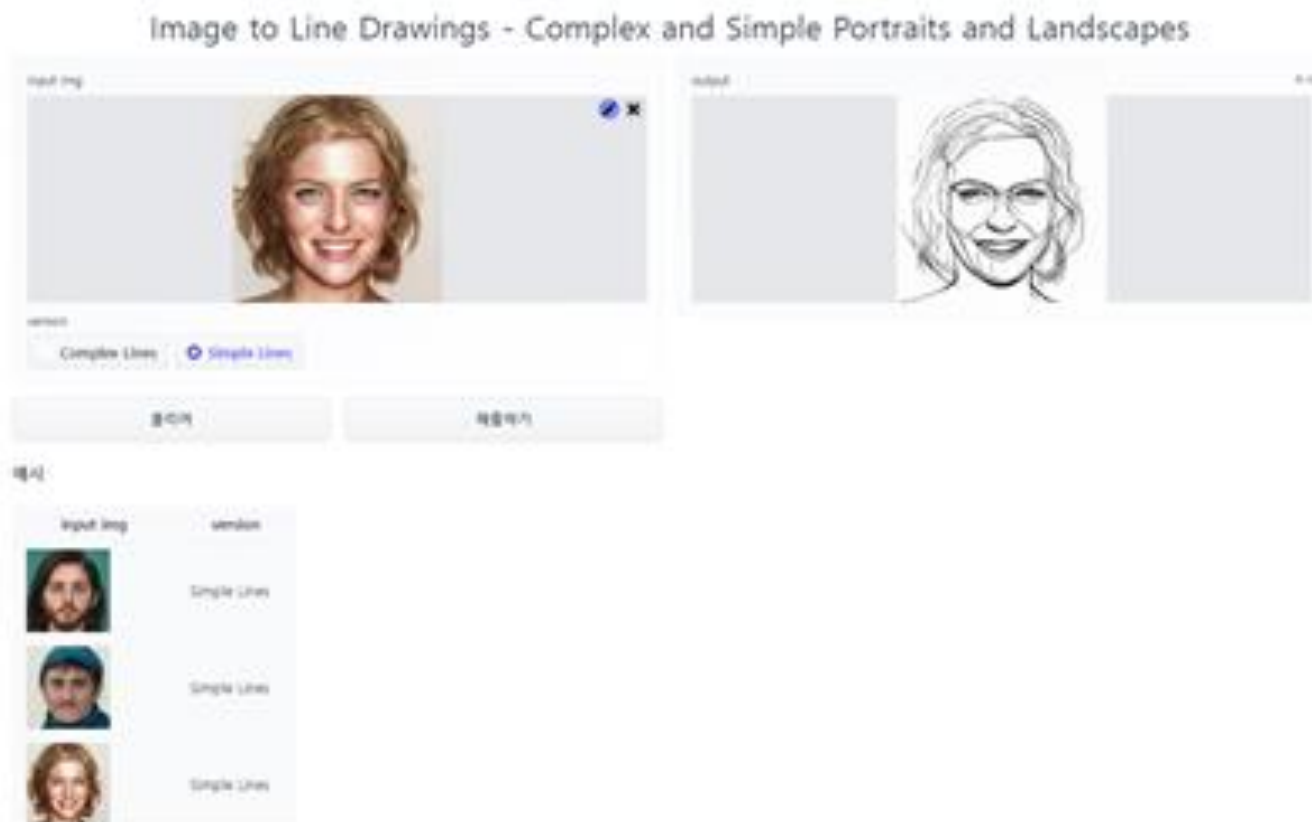
<https://huggingface.co/spaces/awackel/Image-to-Line-Drawings>

[실행 방법] 먼저 예시의 세 번째 이미지를 선택하여 제출하기를 누르면 오른쪽에 실루엣 스케치가 나타난다.



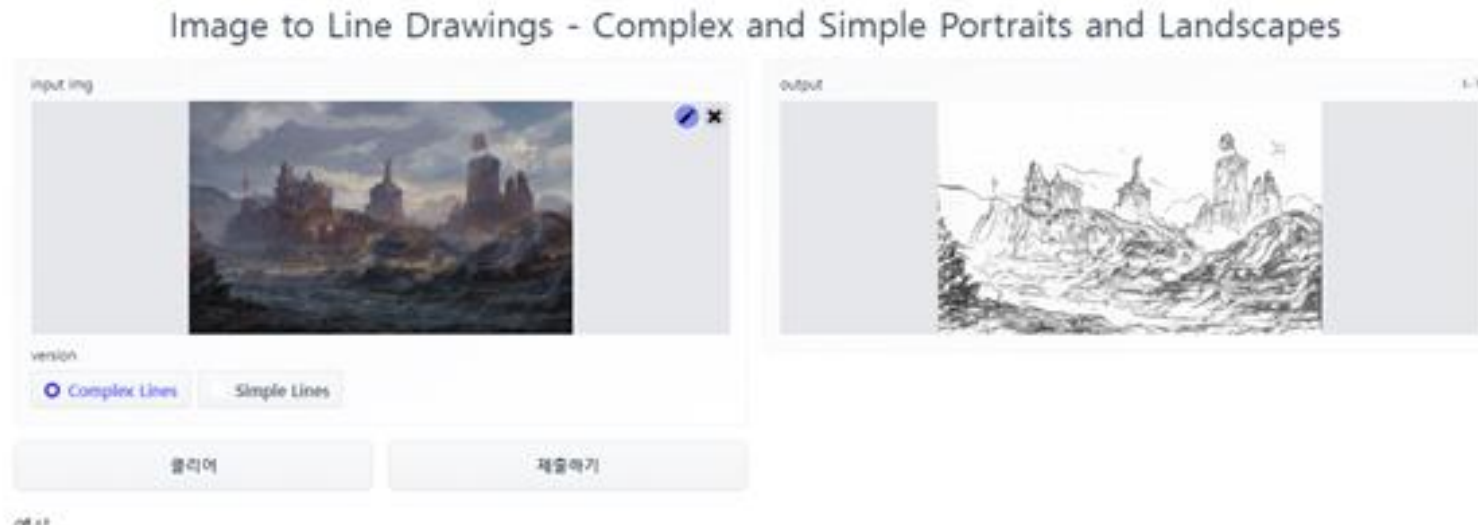
인공지능 실습하기

[실행 결과] 오른쪽에 실루엣 스케치가 나타난다.



인공지능 실습하기

[실행 결과 2] 그 후 "클리어"하고난 후, 다소 복잡한 이미지 중 첫 번째 풍경을 제출하면 다음과 같은 실루엣이 나타난다.



물론 "클리어"를 누른 후 점선 박스를 클릭하여 컴퓨터에 저장된 디렉토리/파일을 찾아 업로드하여 실습할 수도 있다.

인공지능 실습하기

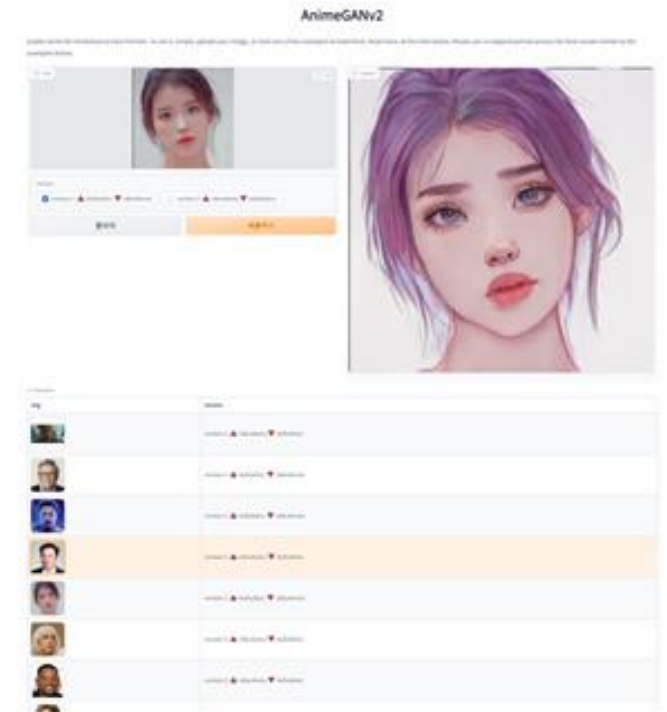
이미지를 애니메이션화 해주는 AI

주어진 이미지의 특징을 캐치하여 애니메이션화하여 보여준다.

<https://huggingface.co/spaces/akhaliq/AnimeGANv2> (클릭)

[실행 방법] 먼저 예시의 다섯 번째 이미지를 선택하여 제출하기를 누르면 오른쪽에 애니메이션화 이미지가 나타난다.

[실행 결과] 오른쪽에 애니메이션화 이미지가 나타난다.



인공지능 실습하기

[실행 결과 2] 오른쪽에 애니메이션화 이미지가 나타난다.



인공지능 실습하기

포켓몬 카드 생성 AI

다양한 포켓몬 카드를 생성한다.

https://huggingface.co/spaces/ronvolutional/ai-pokemon-card_AI

[실행 방법] 제출하기를 누르면 다음과 같은 새로운 포켓몬 카드가 생성된다.



인공지능 실습하기

[실행 결과] 그 결과 다음과 같은 새로운 포켓몬 카드가 생성된다.



인공지능 실습하기

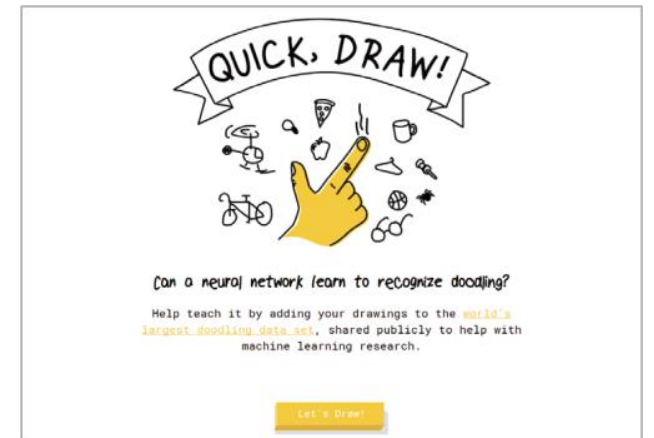
[QuickDraw] 빠르고 정확하게 그림 그리기

신경망이 제시된 단어에 부합하는 그림을 제대로 그렸는지를 판정

<https://quickdraw.withgoogle.com/>

[실행방법]

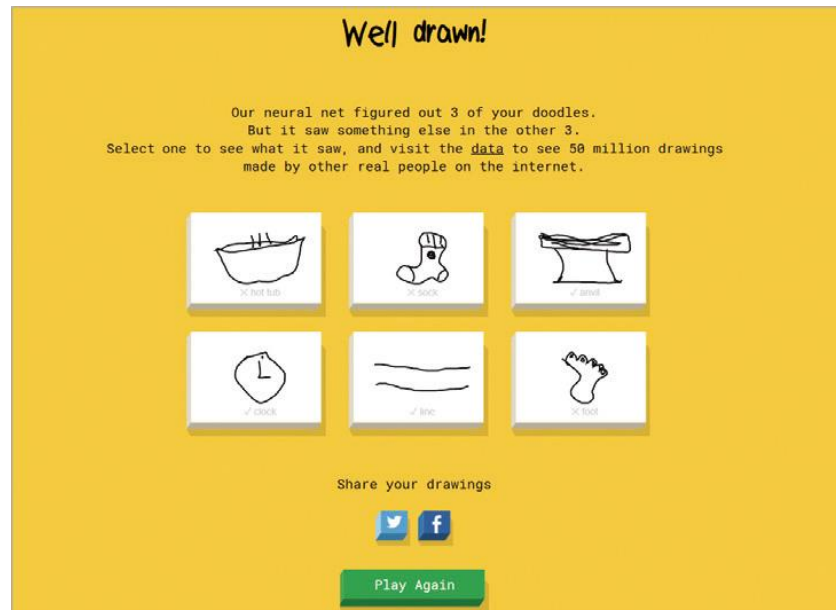
- 시작하기를 누르고 해당하는 그림을 마우스로 그리기
- 각 20초 안에 제시되는 6개 단어에 해당하는 그림 그리기
- 신경망은 계속해서 그린 그림에 대한 단어 맞추기를 시도



인공지능 실습하기

[실행결과]

- 저자의 시도 결과 신경망이 6개 중 3개를 맞춘 것으로 판정
- 그대는 몇 개를 맞추려나?
- 가능하면 빠르게 특징을 잘 표현하기

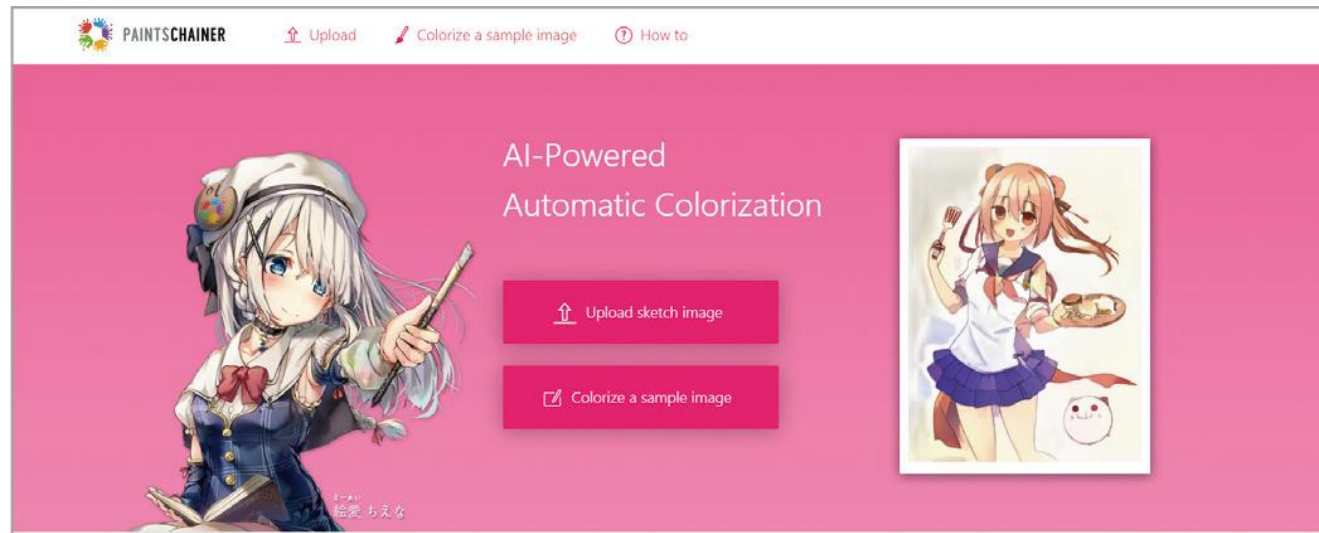


인공지능 실습하기

[Image Colorization] 인공지능 이미지 자동 색칠하기

- 사용자가 살짝만 칠하면 인공지능이 작동
- 이미지 부분에 균형 있게 자동으로 모두 색칠함

https://paintschainer.preferred.tech/index_en.html



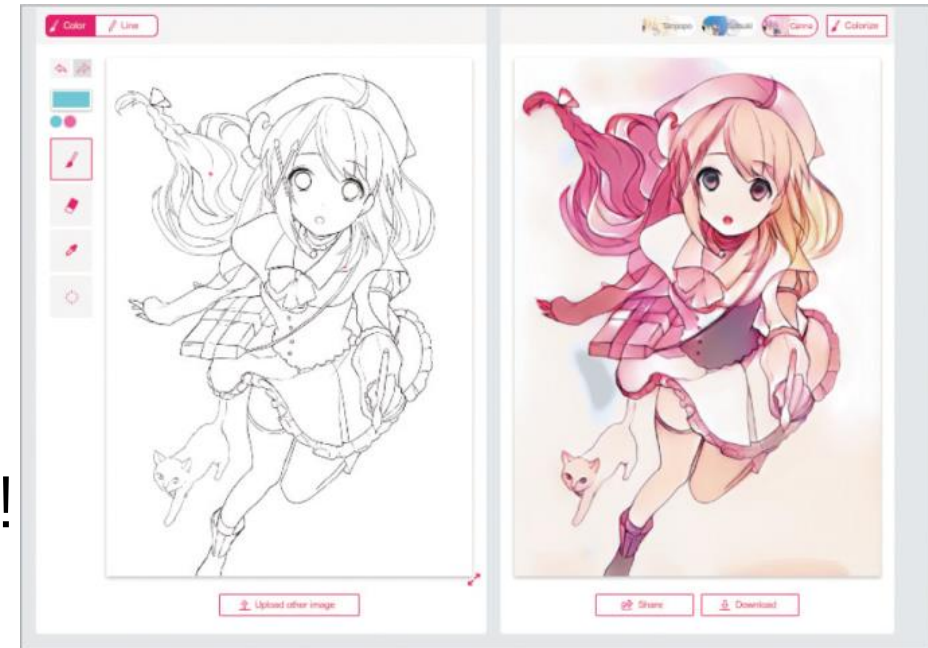
인공지능 실습하기

[실행방법]

- 스케치 데이터를 업로드하거나 샘플 이미지 색칠 가능
- 사용자가 원하는 색을 지정해줄 수 있음

[실행결과]

- 왼쪽 이미지에다 색상과 펜 등의 도구를 선택
- 약간만 터치해도 오른쪽과 같은 이미지가 자동으로 완성!



인공지능 실습하기

잡음이 있는 음향 데이터에서 음질 향상

[Noise Reducer]

(Speech Enhancement Generative Adversarial Network)

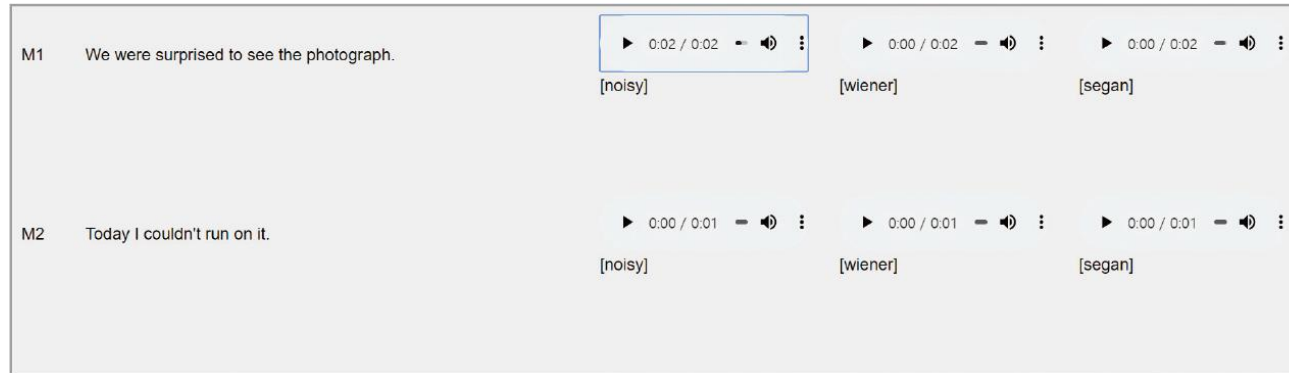
- 주어진 음향 데이터에서 딥러닝 방법으로 잡음을 제거함으로써 음질을 향상시킨다.

<http://veu.talp.cat/segan/>

인공지능 실습하기

[실행방법] 재생 버튼을 눌러 들어보기

[실행결과] 향상된 품질의 음질을 들을 수 있음



인공지능 실습하기

Generative Adversarial Network(GAN)

[GAN playground]

- GAN을 MNIST에 대해 직접 학습시키기
- Generator와 Discriminator가 어떻게 서로를 학습시키는지 확인
- 참고로 MNIST는 손으로 쓴 숫자들로 이루어진 대형 데이터베이스
- (Modified National Institute of Standards and Technology)

<https://reiinakano.com/gan-playground/>

인공지능 실습하기

[실행방법]

- 왼쪽의 각종 하이퍼 파라미터를 바꿔봄
- TRAIN 버튼을 누름

[실행결과] 오른쪽에 실제 학습되는 모습 확인!

