사용자 필적 맞춤형 폰트 생성 서비스

김예진, 이수연, 심규민, 전경구
인천대학교 임베디드시스템공학과
e-mail:yejin8861@naver.com

Custom Handwriting Font Creation Service

Ye-Jin Kim, Soo-Yeon Lee, Kyu-Min Sim, Kyung-Koo Jun
Dept of Embedded System Engineering, In-Cheon University

요 약

한 벌의 한글 글자체를 만드는데 일반적으로 많은 제작 비용과 시간이 소요된다. 따라서 폰트 제작의 어려움을 덜기 위해, 사용자가 대표 글자들을 입력하면 그 글자들의 디자인 특성을 딥러닝 기술을 이용하여 학습한 모델이 나머지 글자들을 자동 생성해주는 시스템 구축한다면 폰트 제작이 훨씬 용이해질 뿐만 아니라 저작권 문제로부터 자유로워질 것이다. 이와 관련된 선행연구를 실행하고 분석해 본 결과 데이터 전처리 과정에서 글자가 잘리거나 크기가 맞지 않아 제대로 된 데이터셋이 구축되지 않는 문제가 있음을 발견하였다. 본 논문에서는 이러한 문제를 해결하기 위해 템플릿에서 자동적으로 글자영역을 추출하고 이미지를 보정하는 전처리 과정과 함께 기존 모델에서 새로운 필터를 추가하여 학습 성능을 높이는 방법을 제안한다. 이를 통해 기존 연구에서 측정된 손실값을 낮춘 결과를 확인했으며 결과적으로 실제 글자체와 더욱 유사한 사용자 맞춤형 글자체를 제공할 수 있을 것이다.

1. 서 론

최근 '개성 표현'이 소비자 트렌드의 주요한 특성으로 부상하면서, '나만의 것'에 대한 수요가 증가하는 추세이다. 이러한 수요는 폰트에 있어서도 동일하게 발생한다. 또한, 저작권 강화로 폰트도 지적재산으로 인정받음으로써, 신규 개발된 글자체를 이용한 사업모델도 가능하다. 하지만 한글 폰트는 영문 로마자 폰트에 비해 제작 노력이 많이 들고 비용이 높다. 영문 폰트는 90여개 글자를 디자인하는 것으로 글자체 한 벌을 만들 수 있지만, 한글 글자체의 경우 11,172개의 글자 디자인이 필요하다. 따라서 한 벌의 글자체를 만들기 위해서는 다수의 디자이너가 장기간 투입되어야 하며, 그에 따른 비용이 높아지게 된다. 하지만 이러한 과정을 딥러닝으로 처리하게 된다면 글자체를 만드는 시간과 비용을 절감시킬 수 있다. 또한, 한글 글자체 제작 생산성을 향상시켜 폰트 시장 자체의 규모를 키울 수 있을 것이다. 딥러닝을 이용한 폰트 생성 서비스에 관한 선행연구로는 조건부 GAN을 사용하여 중국 글자 스타일을 배우는 것을 목표로 하는 zi2zi(Master Chinese Calligraphy with Conditional Adversarial Networks)[1]의 아키텍처를 변형 시켜 한국어 문자 세트를 지원하도록 변경하고 폰트 파일(.ttf)을 생성하는 프로젝트[2]가 존재한다. 하지만 실제로 해당 선행연구를 실행한 결과는 학습의 성능이 폰트로 사용하기에는 부족한 결과였다. 따라서 글자 이미지 전처리 및 딥러닝 모델 구조 변경을 통하여, 선행 연구 결과보다 더욱 성능을 높여, 실생활에서 사용할 수 있는 폰트 파일을 만드는 것을 목표로 한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 전체적인 프로젝트 구조에 대해 기술하고, 3장에서는 기존의 선행연구와 비교하여 개선시킨 사항들과 구체적인 알고리즘에 대해 기술하고 기존 알고리즘과 개선된 알고리즘과의 결과를 비교하며, 마지막으로 4장에서는 결론 및 향후 연구 방향으로 끝을 맺고자 한다.

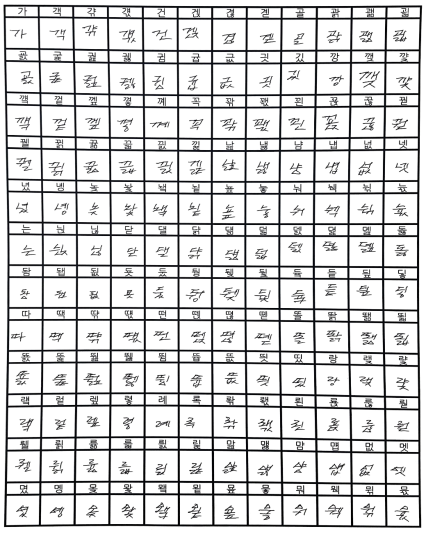
본 논문은 과학기술정보통신부 정보통신창의인재양성사업의 지원을 통해 수행한 ICT멘토링 프로젝트 결과물입니다.

2. 프로젝트 전체 구조

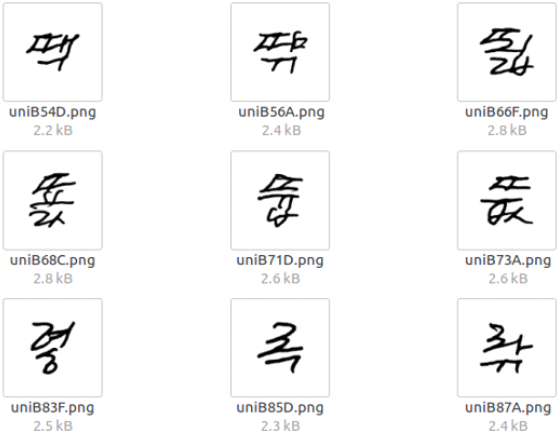
사용자의 글씨 특징과 유사한 폰트를 생성하는 프로젝트의 과정은 다음과 같다.

2-1 crop

우선 사용자가 자음과 모음의 조합으로 구성된 템플릿(그림 1)에 399자의 글자를 작성하면 해당 이미지 파일에서 글씨 영역만을 검출하여 글자별로 잘라낸 뒤, 이진화 작업과 필터를 사용하여 글씨를 보정하고 각 글자의 unicode를 이름으로 한 399자의 이미지 파일을 만들어낸다.(그림 2)



(그림 1) 템플릿



(그림 2) 템플릿의 글자를 한 글자씩 잘라낸 이미지

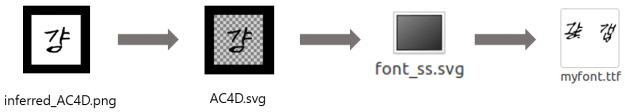
2-2 font2image & package

잘라낸 이미지를 이용해 손글씨 이미지와 해당 글자의 기본이 되는 고딕체 글자를 pair형태로 묶어 이미지 파일을 만든 후, 학습 시 병목현상을 막기 위해 binary format으로 오브젝트 파일을 생성하는 병렬화 과정을 거친다.

2-3 train & inference

GAN[3]을 변형시킨 pix2pix모델[4]을 글자 생성 목적에 맞게 다시 변형한 모델(그림 4)에 전처리 과정을 거친 데이터셋을 이용해 생성자와 판별자의 학습 진행 후, 손글씨 특징에 맞게 399자 외의 나머지 한글 11,172자 이미지를 생성해낸다.

2-4 font생성



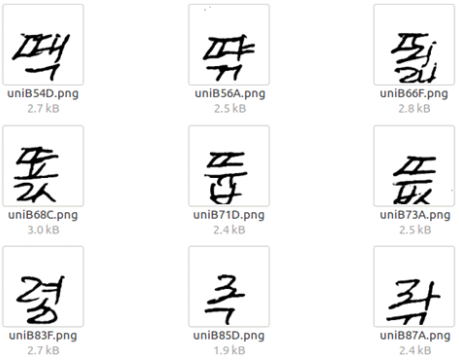
(그림 5) 이미지에서 svg파일을 거쳐 폰트파일이 만들어지는 과정

이러한 글자 이미지 파일들을 모아 nodejs 모듈인 svgicons2svgfont와 svg2ttf모듈을 사용해 최종적으로 워드나 파워포인트에서도 사용 가능한 폰트 파일을 만들어낸다.

3. 개선 사항

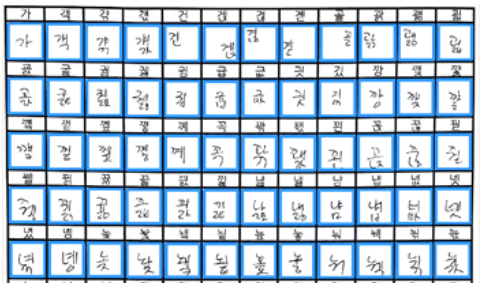
3-1. 데이터 전처리

기존의 오픈소스는 템플릿에서 글자를 잘라내는 crop 과정이 단순히 사용자가 업로드한 이미지에서 가로, 세로 12등분을 하여 그 크기만큼 잘라내는 하드코딩 방식으로 이루어져 있다. 그러나 이러한 코드는 사용자가 업로드한 템플릿이 아주 조금만 비뚤어져도 아래와 같이(그림 6) 글자가 잘리는 문제가 발생할 수 있다. 이렇게 잘못된 데이터셋을 가지고 학습하는 경우 당연히 모델의 성능이 떨어질 수 밖에 없다.

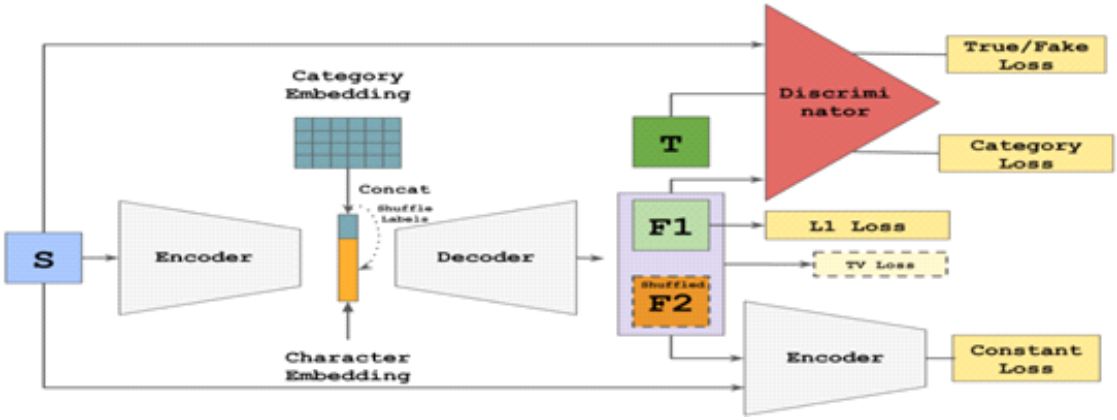


(그림 6) 기존 오픈소스의 crop 과정에서의 문제점

따라서, 이러한 문제를 해결함과 동시에 사용자의 편의성을 증가시키기 위해 우리는 템플릿에서 자동적으로 글자영역을 추출하고 잘라내는 방식을 고안하였다.

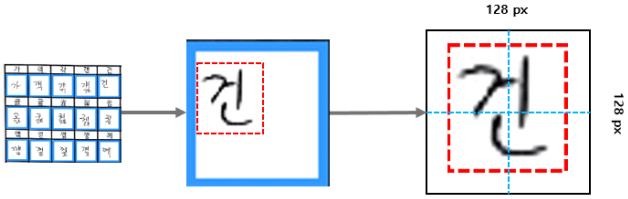


(그림 7) 템플릿에서 각 글자 칸을 검출한 이미지



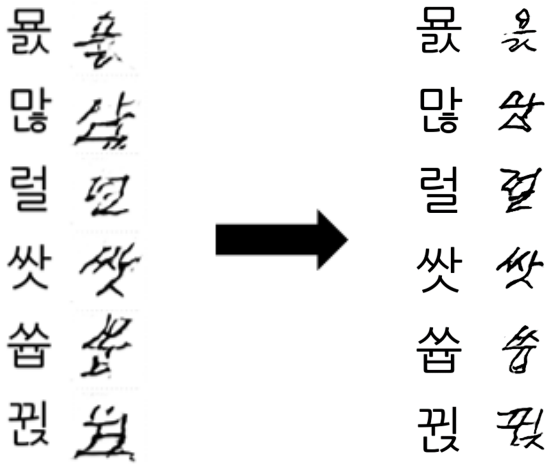
(그림 4) pix2pix모델을 글자 생성 목적에 맞게 변형시킨 모델[1]의 구조

먼저 윤곽선을 검출을 위해 OpenCV의 findcontour 함수를 사용한다. Contour란 동일한 색 또는 동일한 색상 강도를 가진 부분의 가장자리 경계를 연결한 선으로 이미지에 있는 물체의 모양 분석이나 객체 인식 등에 유용하게 활용된다. OpenCV의 findcontour함수는 검정색 배경에서 흰색 물체를 찾는 것과 비슷하므로 윤곽선의 두께를 팽창시켜주는 dilate함수를 이용해 템플릿의 각 글자 칸 경계를 나누는 선을 팽창시킨 뒤 findcontour함수를 적용하면 각 글자 칸 안의 흰색 부분을 검출할 수 있으며(그림 7) 검출된 칸 안에서 또다시 윤곽선(실제 글자)을 찾고 그 윤곽선을 감싸는 최소 넓이의 정사각형을 구하는 OpenCV의 minAreaRect함수를 이용해 글자를 감싸는 사각형 영역을 구한 뒤 128*128 크기의 흰색 바탕에 비율에 맞춰 해당 글자의 크기를 조절한 후 배경의 중심에 오도록 붙여 넣는다.(그림 8)



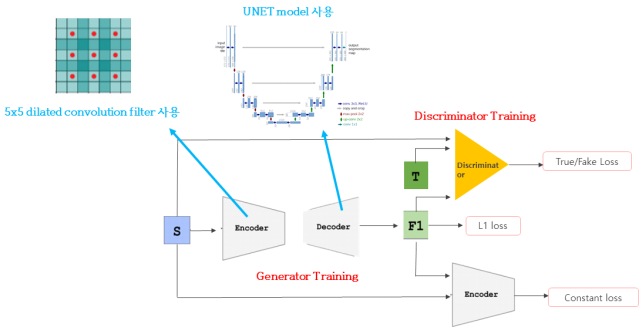
(그림 8) 실제 글자의 위치를 찾아 센터링하는 과정

이렇게 하면 사용자가 정확히 가운데에 글자를 쓰지 않고 치우치게 쓰는 경우와 글자들의 크기가 제각각인 경우 모두를 포함하여 일정한 글자 크기와 위치를 갖게 되어 더욱 정확한 데이터셋이 만들어지게 되고 이는 모델의 성능을 향상시킴에 있어 중요한 역할이 된다.



(그림 9) 기존 전처리 알고리즘을 이용해 만든 데이터로 학습한 결과(좌측)와 변경한 알고리즘을 적용한 데이터로 학습한 결과(우측)

3-2. 필터 추가



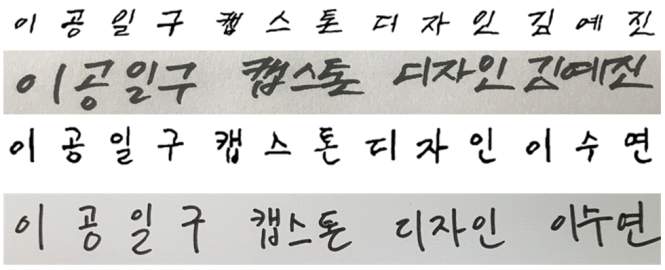
(그림 10) encoder에 dilated convolution filter를 추가한 모델의 구조

기존 모델[1]의 생성자 속 encoder에 사용되는 5*5 필터 대신 dilated convolution filter를 사용하여 글자의 특징점을 더욱 효과적으로 추출하도록 하였다. Dilated Convolution은 확장된 Convolution으로 Convolution layer

에 또 다른 파라미터인 dilation rate을 도입한 것이다. 필터 내부에 zero padding을 추가해 강제로 receptive field를 늘리는 방법으로 (그림 10)의 필터 그림에 있는 빨간점만 weight가 있고 나머지 부분은 0으로 채워진다. 필터가 한 번 보는 영역으로 사진의 feature를 추출하기 위해선 일반적으로 receptive field가 높을수록 좋다. pooling을 수행하지 않고도 receptive field를 크게 가져갈 수 있기 때문에 spatial dimension 손실이 적고 대부분의 weight가 0이기 때문에 연산의 효율이 좋다. 따라서, 기존의 5*5 convolution 대신 Dilated convolution을 사용하게 되면 공간적 특징을 더 잘 유지하는 효과가 있기 때문에 더 정확한 글자의 위치정보를 가져올 수 있게 되며 이는 생성자가 만들어내는 글자가 더욱 실제 손글씨가 비슷하게 학습하는데 도움이 된다. 이로 인해, 판별자가 생성자가 만들어낸 가짜 이미지를 진짜라고 판별하는 cheat_loss의 하락이 두드러졌으며 생성자의 전체적인 loss값들의 합인 g_loss가 기존 5*5 convolution layer를 사용했을 때보다 더 감소하는 결과를 얻을 수 있었다.

(표 1) Dilated convolution layer 변경에 따른 loss값 변화

글씨체	dilation	cheat_loss	l1_loss	g_loss
1	x	0.428	29.12	30.68
1	o	0.199	29.44	**26.35**
2	x	0.689	27.78	29.47
2	o	0.013	27.12	**27.93**



(그림 11) 데이터 전처리 수정과 Dilation을 적용하여 만든 모델의 폰트 생성 결과(위)와 실제 손글씨 비교(아래)

4. 결 론

지금까지 사용자 필적 맞춤형 폰트 생성 서비스에 관한 전반적인 진행 과정에 대해 서술하였다. 그중 기존의 선행 연구[2]의 문제점이라 여겼던 데이터 전처리 과정에서 자동적으로 글자 영역을 추출하고 센터링하는 과정과 기존 모델에 dilated convolution filter를 적용하여 모델의 레이어를 변경한 방법을 통해 전체적인 loss값이 줄어드는 결과를 얻었으며 조금 더 손글씨와 가까운 폰트를 얻을 수 있었다.

향후 연구 계획으로는 이렇게 만들어진 폰트 파일을 더 많은 사용자들이 이용할 수 있도록 폰트 제공 웹 서비스를 구축하여 제공할 예정이다.

참고문헌

[1] https://github.com/kaonashi-tyc/zi2zi
[2] https://github.com/periannath/neural-fonts
[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. In NIPS, 2014.
[4] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. In CVPR, 2017.