공학석사학위논문

CRNN과 BiLSTM을 활용한 동영상 속 한국어 문자 인식

Optical Korean Character Recognition using CRNN with BiLSTM for video analysis

2023 년 2 월

서울대학교 대학원

산업공학과

김 어 진

CRNN과 BiLSTM을 활용한 동영상 속 한국어 문자 인식

Optical Korean Character Recognition using CRNN with BiLSTM for video analysis

지도교수 조 성 준

이 논문을 공학석사 학위논문으로 제출함

2022 년 12 월

서울대학교 대학원

산업공학과

김 어 진

김어진의 공학석사 학위논문을 인준함

2023 년 1 월

위 원 장 이 덕 주 (인)

부위원장 조 성 준 (인)

위 원 이 성 주 (인)

초록

현재 딥러닝 기반 광학문자인식 (OCR) 은 영어, 숫자에 대해서는 성능이 매우 뛰어나다. 그러나 글자 종류가 많은 한글의 특성으로 인해 한글 인식에는 낮은 성능을 보이고 있다. 또한, 대부분의 모델의 경우 동일한 배경을 가진 이미지에 대해서 높은 성능을 보이며 다양한 배경이 존재하는 동영상들에 대해서는 낮은 성능을 보인다. 본 연구에서는 한글 글자 종류 압축, 학습 데이터 조합, 모델 구조의 변화 및 결과 후처리를 진행하여 한글 인식 성능 향상을 시도하였다. 제시한 모델은 현재 가장 성능이 좋은 한글 OCR 오픈소스 라이브러리인 EasyOCR의 결과보다 매우 뛰어난 결과를 보였다. 본 연구를 통해 모델의 한글 인식의 성능을 향상하였다. 추가로 소수의 실패 사례들을 분석하여 향후 연구방향에 대하여 제시한다.

주요어: OCR, 한글인식, 동영상 분석, 실사데이터, CRNN, BiLSTM

학번: 2021-28110

목차

표 목차

그림 목차

제 1 장 서론

최근 컴퓨팅 성능의 향상과 더불어 매일 방대한 데이터들이 생성됨에 따라 인공지능을 다양한 분야에 적용하는 연구가 늘어나고 있으며, 특히 이미지를 분석하는 컴퓨터 비전 분야의 경우 다양한 분야에서 응용될 수 있다. 동영상 플랫폼의 경우 연속된 이미지들의 집합체로 이루어져 있기에 컴퓨터 비전 분야의 기술을 통해 다양한 연구 및 분석이 가능하다.

인터넷 성능 및 스마트폰 성능 향상으로 인해 점차 동영상 플랫폼은 현대인의 취미, 정보처 등 다양한 형태로 삶에 스며들었고 이에 따라 동영상 분석의 중요성이 대두되고 있다. 대표적으로 패션분야에서 동영상 분석을 효과적으로 활용할 수 있다. 패션은 유행에 매우 민감하다는 특성을 가지고 있다. 따라서 패션에 대한 분석을 진행하고자 할 때는 지속적으로 변하는 유행을 추적할 필요가 있으며, 이러한 유행에 대한 정보를 가장 잘 반영하는 매체는 영상 플랫폼 “유튜브” 이다.

패션에서의 유행은 기준에 따라 크게 두 가지로 분류된다. 첫째, 스타일이 유행하는 경우이다. 스타일 유행은 사회, 문화 혹은 환경적인 요소에 영향을 받는다. 예시로 코로나의 유행이 본격적으로 시작한 2021년에는 외출이 적어진 생활 양식으로 인해 편한 스타일이 유행을 하기 시작하였으며 또한, 마스크 스트랩 등 기존에 존재하지 않던 새로운 아이템이 패션 요소로 인식되기 시작했다. 이러한 스타일 유행은 유튜브에서 이미지로서 반영이 되기에 유행을 분석하기 위해서는 이미지 자체의 옷의 형태와 색깔 등 세부 요소를 분석할 필요가 있다.

둘째, 브랜드가 유행하는 경우이다. 브랜드 유행은 브랜드가 대중에게 노출되는 정도와 대중이 브랜드에게 기대하는 이미지에 영향을 받는다. 예시로 유명인이 특정 브랜드의 의류를 착용한 사실이 화제가 되거나 혹은 특정 단체, 세대에서의 브랜드의 입소문이 퍼지는 경우를 들 수 있다. 이러한 브랜드 유행은 유튜브에서 주로 텍스트로서 반영이 되기에 분석하기 위해서는 유튜브 동영상의 이미지 속의 텍스트를 분석할 필요가 있다. 표 1.1은 유튜브에서 브랜드 유행을 분석한 결과이다. 해당 표는 패션 관련 유튜브 영상에서 각 브랜드가 노출된 정도를 각각 2022년 7월 2일, 2022년 7월 9일을 시작일로 일주일 동안의 노출도를 분석하여 노출도가 가장 높았던 8개의 브랜드를 정리한 결과이다.

Brand	Exposure	Brand	Exposure
NIKE	50	NIKE	46
BALENCIAGA	44	BALENCIAGA	35
SUPREME	31	SUPREME	29
SALOMON	27	CONVERSE	27
STUSSY	25	VANS	27
VANS	24	SALOMON	23
ADIDAS	24	STUSSY	20
NEW BALANCE	24	LEMAIRE	20

표 1.1 브랜드 노출도 분석 결과 (2022. 07. 02, 2022. 07. 09)

그림 1.1과 같이 동영상의 텍스트는 이미지 형태로 이루어져 있으며 이를 텍스트 데이터로 전환하여 분석하기 위해서는 Optical Character Recognition (OCR) 기술이 필요하다. OCR은 텍스트가 적혀 있는 이미지 형태의 데이터에서 텍스트를 얻어 내는 기술로써 동영상 분석과 함께 시각장애인을 위한 책 읽기 서비스, 문서파일 및 인쇄물 판독 등 여러 분야에서 활용될 수 있는 중요한 기술이다. 이에 많은 OCR 관련 연구가 진행되었으며 그에 따른 성과를 냈다.

그림 1.1 유튜브 동영상 내 프레임 예시[1]

그러나 선행 OCR 연구는 동영상 분석에 적용하기에 있어 두 가지 문제점을 보인다. 첫째는 대부분의 선행 연구에서 제안한 모델들이 영어와 숫자 및 특수문자에만 국한된 성능을 보였으나 이를 한글에 적용하는 데 어려움을 겪고 있다는 점이다. 한글은 영어에 비해 라벨링에 있어서 어려움이 있다. 영어의 경우 문자를 구성하는 글자의 종류가 총 알파벳 26개이다. 따라서 최종적으로 OCR 모델은 영어 문자를 인식하기 위해 26개의 라벨 중 하나로 분류를 진행한다. 그러나 한글의 경우 문자가 초성, 중성, 종성 총 3음절의 조합으로 이루어져 있으며 각 음절의 종류의 수는 19개, 21개, 27개이기에 한글 문자를 구성하는 글자의 종류의 수가 총 19 * 21 * (27+1) = 11172개다. 결

국 OCR 모델을 이용해 한글 문자를 인식한다는 것은 하나의 글자를 11172개의 라벨 중 하나로 분류하여야 한다는 뜻이고 이는 한글 OCR 모델 성능 발전을 어렵게 하는 이유가 된다.

또한 한글 OCR 모델 학습을 위한 데이터가 현저하게 부족하여 많은 연구자는 학습에 어려움을 겪었다. 위에서 언급한 대로 영어에 비해 한글 문자를 나타내기 위한 라벨의 수가 훨씬 많기에 학습 데이터의 부족은 치명적이다. 하지만 최근 한국지능정보사회진흥원에서 운영하는 학습 데이터 공유 플랫폼 AIHub에서 한글 OCR 모델 학습을 위한 데이터를 구축했다. 데이터는 인쇄체, 손글씨 및 실사 데이터로 이루어져 있다.

둘째는 기존 OCR 모델 연구의 경우 정형화된 배경의 문자를 인식하는 데 집중하여 연구가 진행되었기에 범용성이 부족하다는 점이다. 현재 OCR 모델은 영수증 인식, 번호판 인식 등과 같이 일정한 형태의 배경 내에서 인식을 진행한다. 하지만 동영상 내 텍스트의 경우 다양한 배경이 존재하며 글자의 색깔 및 폰트 또한 다양한 분포를 보이기에 기존 OCR 모델을 활용하여 동영상 분석을 시도했을 때 낮은 성능을 보인다.

본 연구에서는 기존 선행 연구가 진행된 OCR 모델을 기반으로 모델 변형, 학습 데이터 세팅 및 데이터 후처리를 통해 위의 문제를 완화하여 한글 OCR 모델 성능 향상을 이루었다. 인식하고자 하는 한글 글자의 종류의 수를 압축하여 한글 OCR 모델에 최적화된 한글 라벨을 찾았으며 위에서 언급한 인쇄체, 손글씨 및 실사 데이터를 다양하게 조합하여 학습을 진행함으로써 동영상 데이터 분석에 최적화된 OCR 모델을 구현하였다. 또한, 한글 인식에 최적화된 모델 구조를 찾아냈으며 인식한 결과에 다양한 후처리를 적용하여 성능을 향상시켰다.

본 논문은 서론, 선행 연구, 모델 구조, 해법, 실험 결과, 결론 총 6장으로 이루어

져 있다. 제1장 서론에서는 본 연구의 배경과 연구개발의 목적 및 필요성을 언급하였다. 제2장 선행 연구에서는 OCR 모델 선행 연구 및 관련 연구에 관해 서술한다. 제3장 해법에서는 전체 모델의 구조와 OCR 모델을 한글에 최적화하기 위한 한글 라벨링 방법, 학습 데이터 세트 및 변형한 모델 구조에 관해서 서술한다. 제4장 실험 결과에서는 실험 환경 및 성능 평가 기준, 실험 결과에 관해서 서술하며 마지막 제5장 결론에서는 본 연구의 결론과 향후 연구 방향에 대해서 제시한다.

제 2 장 선행연구

2.1 CNN (Convolutional Neural Network)

Convolutional Neural Network (CNN) 은 다차원 데이터를 처리하기 위해 등장했다. 대표적으로 이미지 형태의 데이터를 분석하기 위해 사용하는데 데이터의 공간정보를 유지하며 학습을 진행한다는 특징을 가지고 있다. CNN은 Convolution Layer와 Pooling Layer가 반복적으로 쌓여 있는 구조로 이루어져 있다. Convolution Layer에서 Convolution 연산을 통해 공간정보를 유지하며 이미지의 특징을 추출하며 특징을 강화하고 이미지의 크기를 축소하는 Pooling Layer의 Pooling 연산을 반복하여 이미지에서의 패턴, 즉 추상화된 특징을 얻는다[2].

2.2 RNN (Recurrent Neural Network)

Recurrent Neural Network (RNN) 은 Sequence 데이터를 처리하기 위해 등장했다. RNN의 주요한 특징 중 하나는 내부에 순환구조가 들어있다는 것이다. RNN은 학습을 진행하면서 입력되는 모든 데이터의 정보를 요약하는 Hidden State를 가지고 학습을 진행한다. 데이터가 입력될 때마다 Hidden State에 정보를 전달하게 되므로 Sequence로 구성된 데이터를 처리하는 데 효과적이며 특히 음성, 혹은 텍스트의 앞뒤 성분을 파악하는 데 높은 성능을 보인다[2].

그러나 초기 RNN의 경우 Sequence의 길이가 길어질수록 예측하려는 지점의 값과 참고하려는 정보 사이의 거리가 멀수록 역전파를 진행하는 과정에서 기울기가 줄어드는 기울기 소실 문제를 보였다. 그림 2.1과 같이 입력 Sequence의 요소들을 앞에서부터 하나씩 입력을 하여 연산을 진행하기에 앞에서 거리가 멀어질수록, 즉 Sequence의 길이가 길어질수록 앞의 정보들은 지속되는 연산들로 인해 희석된다..

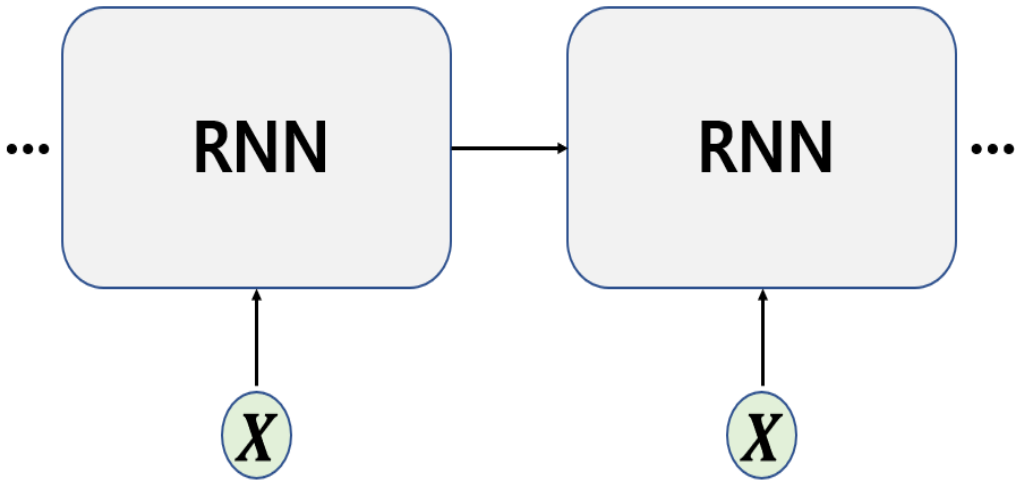

그림 2.1 RNN 모델 구조

기울기 소실 문제를 해결하기 위해 등장한 모델이 LSTM이다. LSTM은 기존의 RNN 구조에 Cell State라는 새로운 흐름을 도입하여 앞에서 입력되었던 정보들에 대한 특징을 추가적으로 전달하여 연산을 진행한다. 그림 2.2와 같이 기존에 진행되던 Sequence들의 흐름에 Cell State라는 흐름을 추가로 도입하여 각 연산마다 입력 Sequence의 앞 부분에 있는 정보들의 반영 여부를 결정하여 기울기 소실 문제를 해결한다[2].

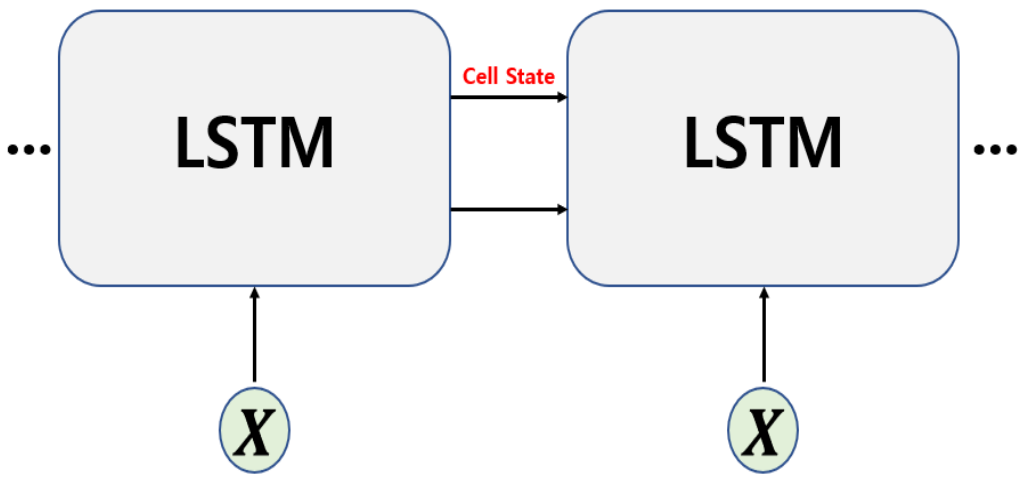

그림 2.2 LSTM 모델 구조

LSTM 모델의 경우 기울기 소실 문제를 해결하였으나 RNN과 LSTM 모델 모두 입력 Sequence를 시간 순서로 입력 받기 때문에 각 연산마다 직전 입력 값, 즉 앞부분의 입력 값들을 기반으로 연산을 진행한다는 한계가 존재한다. 따라서 그림 2.3과 같이 기존의 순방향으로만 진행되는 연산에 역방향 연산을 추가하여 직전 앞부분의 입력 값만이 아니라 뒤에 입력 값 또한 참조하여 연산을 진행하도록 만든 BiLSTM 모델이 등장하였다.

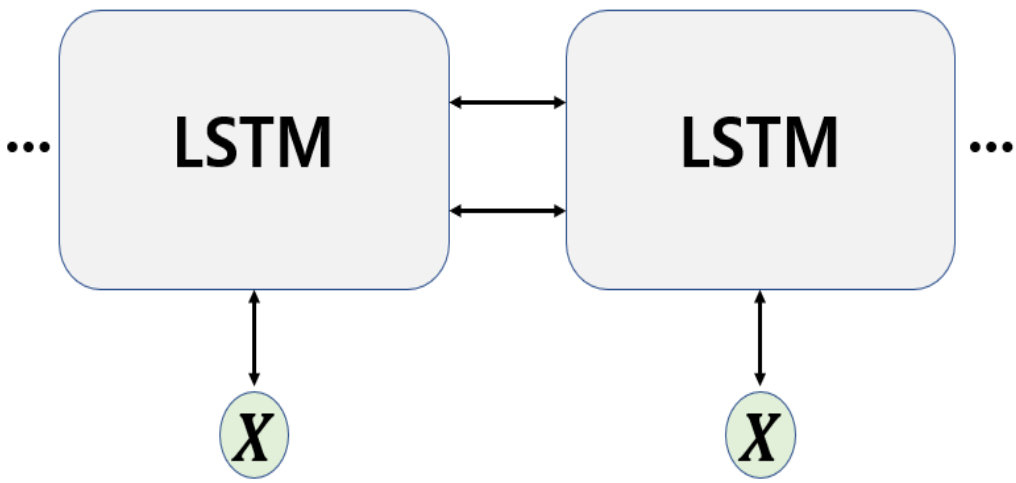

그림 2.3 BiLSTM 모델 구조

2.3 STN (Spatial Transformation Network)

CNN은 이미지 분류 모델에 있어 좋은 성능을 보였으나 회전되어 있는 혹은 휘어져 있는 이미지같이 Normalize 되지 못한 이미지에 대해서는 좋은 성능을 보이지 못한다. 이러한 문제를 해결하기 위해 STN (Spatial Transformation Network) 은 등장했다. STN은 기존의 이미지가 더 좋은 Spatial Invariance 능력을 갖추게 하도록 이미지의 특정 부분을 자르고 변환해서 그 부분만 떼어내는데, 이 과정에서 변환을 위해 필요한 Parameter를 학습한다[3].

2.4 Seq2Seq and Attention

Sequence-to-Sequence (Seq2Seq) 알고리즘은 어떤 Sequence 들을 다른 Sequence로 Mapping 하는 알고리즘이다. OCR 모델은 이미지를 부분별로 잘라 Sequence로 넣어서 텍스트 Sequence를 인식하기 때문에 Seq2Seq 알고리즘이다. 초기 Seq2Seq 알고리즘의 경우 일반적으로 입력 Sequence를 하나의 고정된 크기의 벡터로 압축하고, 이 벡터를 통해서 출력 Sequence를 만들었다[4].

그러나 이러한 방법은 동영상 분석에 적용하기에 있어서 문제를 보인다. 바로 하나의 고정된 크기의 벡터에 모든 정보를 압축하기 때문에 정보 손실이 발생한다는 점이다. 차량 번호판, 영수증과 같이 일정한 크기의 텍스트를 가지고 있는 이미지에 대해서 고정된 크기의 벡터로 압축하더라도 정보 손실이 발생하지 않도록 모델이 학습된다. 하지만 동영상 속의 텍스트의 경우 다양한 크기를 가지고 있으며 이를 일정한 크기의 벡터로 모두 압축시켰을 때 텍스트의 크기에 따라서 정보 손실이 발생하게 된다.

이를 개선하기 위해 등장한 새로운 모델이 Attention이다. Attention 모델은 하나의 고정된 크기의 벡터에 모든 정보를 압축하지 않으며 출력 Sequence의 한 부분을 예측할 때, 매 시점 전체 입력 Sequence를 참조하며, 해당 시점에서 예측해야 하는 부분과 연관이 있는 입력 Sequence의 부분에 더 높은 가중치를 두어 예측을 진행하여 Seq2Seq 알고리즘의 성능을 향상시켰다.

대표적인 Attention 모델 Scaled Dot-Product Attention 모델의 작동 과정을 식으로 나타내면 다음과 같다.

$$\text{Attention}(\text{Q}, \text{K}, \text{V}) = softmax\frac{\text{Q}K^T}{\sqrt{D_K}} \quad (1)$$

Q, K, V는 각각 입력에 특정 가중치를 곱하여 계산을 한 행렬들이며 이 행렬들을 이용하여 최종 예측 값을 계산하게 된다. Attention 모델은 Q, K, V를 도입하여 입력 값에 대한 정보를 반영하고 및 전체 입력 Sequence에서 해당 시점의 입력 값과 연관이 있는 입력 값의 가중치를 높여 계산을 하도록 함으로써 효과적으로 전체 입력 Sequence를 반영한 값을 출력하도록 한다[5].

2.5 CRNN (Convolutional-Recurrent Neural Network)

OCR모델은 이미지를 다루는 컴퓨터 비전 모델이면서 동시에 Sequence를 예측하는 모델이기에 이미지를 벡터들로 추상화하는 CNN모델과 Sequence 데이터를 다루는 RNN 모델의 특징을 모두 필요로 했다. 따라서 초기에 제안된 OCR 모델은 CNN과 RNN이 결합한 Convolutional-Recurrent Neural Network (CRNN) 이다[6]. CRNN은 텍스트 이미지에서 CNN Feature를 뽑아낸 뒤에 이 Feature들을 RNN의 Input Sequence로 넣어 텍스트를 추출한다.

CRNN 이후 성능 향상을 위해 다양한 변형모델이 등장하였다. 예를 들어, 텍스트 이미지를 Normalize 하기 위해 STN을 도입하여 CNN의 특징 추출 성능을 향상시킨 모델이 등장하였으며, 문제 Sequence 예측의 성능을 향상시키기 위해 Attention을 도입하는 모델이 등장하였다[3], [5], [7], [8].

CRNN을 구성하는 CNN과 RNN은 시간이 갈수록 다양한 모델이 등장하고 있다. 이와 관련하여 OCR 모델의 성능이 가장 뛰어난 CNN과 RNN의 모델 조합을 실험을 통해서 찾아낸 연구가 있었다[9].

제 3 장 제안하는 모델

3.1 모델 구조

본 연구에서 사용하는 OCR 모델의 구조는 다음 그림 3.1과 같이 Transformation Stage, Feature Extraction Stage, Sequence Modeling Stage, Prediction Stage로 총 4단계의 Stage로 이루어져 있다[9].

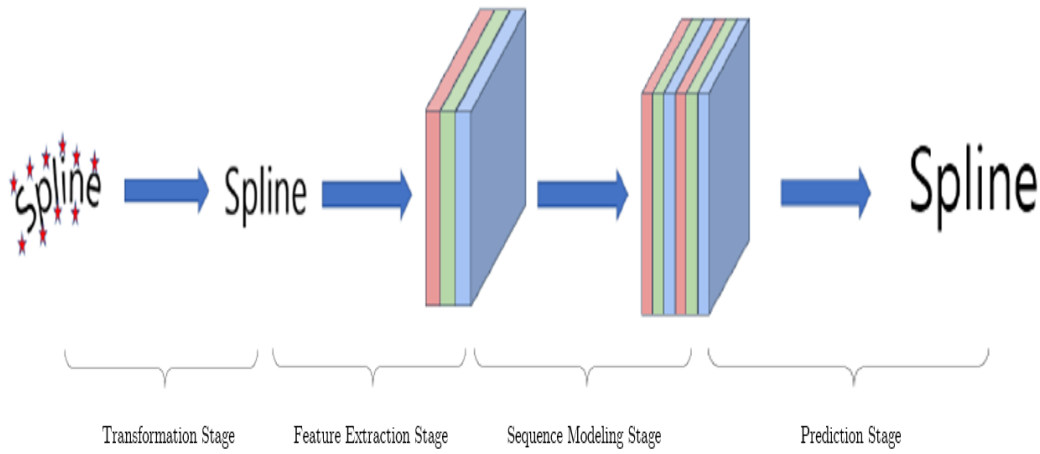

그림 3.1 OCR 모델 구조

3.1.1 Transformation Stage

Transformation Stage의 Module은 Input 이미지를 Normalized 이미지로 변환한다. 일반적인 동영상 속의 텍스트 혹은 일반 이미지의 텍스트들은 휘어져 있거나 혹은 회전되어 있는 경우가 많다. 이들을 모두 학습시키기 위해서는 다양한 형태의 방대한 양의 학습 데이터가 필요하다. 이를 효율적으로 학습시키기 위해서 STN의 변형 모델인 Thin-Plate State (TPS) 모델이 등장하였다[10].

TPS 모델은 이미지 속의 텍스트를 나타내는 기준이 되는 점들을 바탕으로 Spline 변형을 진행하여 직사각형의 형태의 이미지로 반듯하게 나타나도록 한다. Transformation Stage의 예시로 그림 3.2의 왼쪽 휘어진 모양의 “Spline" 텍스트 이미지를 Input으로 넣어 Stage를 진행했을 때 이미지의 빨간 별들을 기준으로 Spline 변형을 진행하여 오른쪽의 바로 선 모양의 “Spline" 텍스트 이미지를 출력한다.

그림 3.2 Transformation Stage 예시

3.1.2 Feature Extraction Stage

Feature Extraction Stage의 기능은 CNN이 담당하고 있으며 Input으로 들어오는 이미지를 추상화하여 Feature를 얻어 내는 것이다. 각 Feature의 부분은 이미지의 각 부분의 정보를 담고 있는 Feature가 되며 이는 이미지의 각 부분의 Text를 예측하는 부분의 Input으로 사용된다. 기존 연구에서는 CNN 모델로 VGG, ResNet 모델을 선택하여 실험을 진행하였다[9].

VGG 모델은 네트워크의 깊이를 증가시킴으로써 성능을 높인 모델이다[11]. VGG 모델 이전의 대표적인 CNN 모델이었던 Alexnet은 다양한 커널을 사용하여 Convolution 연산을 수행하였으며 크기가 큰 커널을 사용하기 때문에 네트워크의 깊이를 높였을 때 연산량이 높아지는 문제가 발생하였고 이로 인해 깊은 네트워크를 구축할 수 없었다[12]. 그러나 VGG 모델의 경우 커널의 크기를 모두 3 * 3으로 고정하여 네트워크의 깊이를 깊게 만들었으며 이는 기존 Alexnet에 비해 높은 성능을 보였다. 기존 연구에서 사용한 VGG 모델은 표 3.1과 같다[9].

Layers	Configurations		Output
Input	Image		100 * 32
Conv1	c: 64	k: 3 * 3	100 * 32
Pool1	k: 2 * 2	s: 2 * 2	50 * 16
Conv2	c: 128	k: 3 * 3	50 * 16
Pool2	k: 2 * 2	s: 2 * 2	25 * 8
Conv3	c: 256	k: 3 * 3	25 * 8
Conv4	c: 256	k: 3 * 3	25 * 8
Pool3	k: 1 * 2	s: 1 * 2	25 * 4
Conv5	c: 512	k: 3 * 3	25 * 4
BN1	Batch normalization		25 * 4
Conv6	c: 512	k: 3 * 3	25 * 4
BN2	Batch normalization		25 * 4
Pool4	k: 1 * 2	s: 1 * 2	25 * 2
Conv7	c: 512 s: 1 * 1	k: 3 * 3 p: 0 * 0	25 * 1

표 3.1 VGG 모델 구조

ResNet 모델은 VGG 모델보다 네트워크의 깊이를 더욱 증가시키면서 동시에 Residual Connection을 도입하여 성능을 높인 모델이다[13]. 앞선 VGG 모델로부터 네트워크의 깊이가 깊어질수록 모델의 성능이 향상된다는 연구 결과를 얻을 수 있었다. 그러나 계속해서 네트워크를 깊게 만들었을 때, 과적합 문제로 인해 오히려 성능이 떨어지는 결과를 보였다. ResNet 모델은 네트워크를 더욱 깊게 만들기 위해 Residual Connection을 도입하였다. ResNet 모델은 그림 3.3과 같이 연산이 진행되는 Layer들을 일정 층으로 나눈 뒤에 각 층마다의 입력 값을 Convolution 연산이 진행되어 출력된 값에 더해서 출력하도록 모델을 설계하였으며, 입력 값을 출력 값에 더하도록 하는 연결이 Residual Connection이다. 기존 연구에서 사용한 ResNet 모델은 표 3.2과 같으며 ResNet 모델을 CNN 모델로 사용한 모델이 가장 높은 성능을 보였다[9].

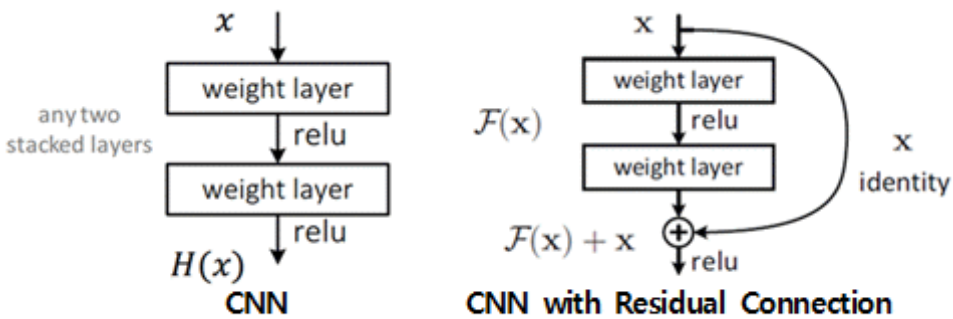

그림 3.3 CNN with Residual Connection

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 32 k: 3 * 3	100 * 32
Conv2	c: 64 k: 3 * 3	100 * 32
Pool1	k: 2 * 2 s: 2 * 2	50 * 16
Block1	[c: 128, k: 3 * 3 c: 128, k: 3 * 3] * 1	50 * 16
Conv3	c: 128 k: 3 * 3	50 * 16
Pool2	k: 2 * 2 s: 2 * 2	25 * 8
Block2	[c: 256, k: 3 * 3 c: 256, k: 3 * 3] * 2	25 * 8
Conv4	c: 256 k: 3 * 3	25 * 8
Pool3	k: 2 * 2 s: 1 * 2 p: 1 * 0	26 * 4
Block3	[c: 512, k: 3 * 3 c: 512, k: 3 * 3] * 5	26 * 4
Conv5	c: 512 k: 3 * 3	26 * 4
Block4	[c: 512, k: 3 * 3 c: 512, k: 3 * 3] * 3	26 * 4
Conv6	c: 512 k: 2 * 2 s: 1 * 2 p: 1 * 0	27 * 2
Conv7	c: 512 k: 2 * 2 s: 1 * 1 p: 0 * 0	26 * 1

표 3.2 ResNet 모델 구조

3.1.3 Sequence Modeling Stage

Sequence Modeling Stage는 앞선 Feature Extraction Stage에서 추출한 Feature를 새로운 Feature의 Sequence로 변환하는 기능을 담당한다. RNN이 담당하고 있으며 기본적인 RNN 모델을 사용할 시에 문맥의 정보가 소실될 수 있다는 단점이 있다. 따라서 기존 연구에서는 기본적인 RNN 모델의 정보 소실 문제를 해결함과 동시에 예측 시점 이전의 정보만이 아니라 이후의 정보까지 참조하는 BiLSTM 모델을 RNN 모델로 사용하여 실험하였고, 그 결과 BiLSTM 사용 모델이 더 높은 성능을 보였다. 따라서 본 연구에서는 RNN 모델로 BiLSTM 모델을 사용한다[6], [9].

3.1.4 Prediction Stage

Prediction Stage는 위의 Sequence Modeling Stage에서 추출한 Feature Sequence를 이용하여 문자 Sequence를 예측한다. 본 연구의 모델에서는 Attention을 이용하는 Prediction Model을 사용한다. Attention 모델은 Input Sequence의 모든 부분의 Feature에 따라 가중치를 두어 참고하여 예측한다. 이러한 특징은 곧 OCR 모델이 문자 단위의 예측 기능을 향상시키는 주요한 요인이 된다[8].

3.2 한글 라벨링

한글은 한 글자가 초성, 중성, 종성으로 이루어져 있으며 이들을 조합하여 만들 수 있는 경우의 수는 11172개이다. 이들 모두를 분류하여 예측하는 것은 모델을 무거워지게 하고, 성능 향상에도 큰 걸림돌이 된다. 또한 11172개의 글자는 일반적으로 모두 사용되지 않으며 이들 중 소수의 글자들이 주로 사용된다. 한국표준협회에서 한국 산업 규격으로 지정한 한국어 문자 집합 “KS X 1001" 에는 11172개의 글자 중 자주 사용되는 2350개의 글자가 정리되어 있다[13]. 본 연구에서는 11172개의 라벨이 아닌 앞서 언급한 “KS X 1001" 의 2350개의 라벨을 이용하도록 모델을 구현하였다.

3.3 학습 데이터 세트

본 연구에서는 최근 AIHub에서 공개한 한국어 글자체 이미지를 가지고 학습을 진행하였다. 공개한 데이터는 손글씨, 인쇄체, 실사 데이터 총 3가지의 종류가 있으며 본 절에서는 이들에 대한 자세한 설명과 더불어 실험 1의 학습에 사용할 데이터 세트에 대해서 서술한다[15].

3.3.1 손글씨

손글씨는 다양성을 확보하기 위해 성별 및 연령층별로 손글씨 작성 인력을

확보하여 개인이 직접 작성 제작한 이미지와 텍스트로 이루어진 데이터 세트이다. 같은 글씨를 개개인이 손으로 적을 때 다양한 모양의 글자가 나오게 된다. 동영상이나 일상에서 사용되는 글자는 손으로 적힌 글자가 많으며, 편집을 통해 변형된 글자들 또한 일반적인 글자체로는 학습이 어려울 수 있기에 손글씨 데이터 세트를 학습 데이터로 이용한다면 이러한 글자들의 특징을 학습하는 데 도움을 줄 수 있을 것으로 기대된다.

3.3.2 인쇄체

인쇄체 데이터 세트는 가장 많이 활용하는 폰트(글자체) 50종을 선정하여 한글 11172자가 해당 글씨체로 적혀 있는 이미지와 텍스트로 이루어진 데이터 세트이다. 일반적으로 동영상에 적혀 있는 텍스트 및 실생활의 이미지에 적혀 있는 텍스트들은 하나의 글씨체가 아닌 다양한 글씨체로 적혀 있다. 인쇄체 데이터 세트를 학습 데이터로 이용한다면 글씨체의 다양함으로 인한 OCR의 어려움을 완화시킬 수 있을 것으로 기대된다.

3.3.3 실사 데이터

실사 데이터는 일상에서 쉽게 볼 수 있는 간판, 상표, 교통표지판 등의 한글이 들어있는 이미지 10만 장과 이미지 내 한글 위치와 텍스트로 이루어진 데이터 세트이다. 실사 데이터는 이미지 내 한글 위치에 맞춰서 자른 후에 학습 데이터로 활용하였다. 앞서 언급한 두 데이터 세트를 통해 다양한 글자의 형태를 학습할 수

있으나 동영상의 글자 혹은 일상 속의 글자들은 글자의 형태와 다양한 배경이 함께 존재하기에 문자 인식이 어렵다. 따라서 주변의 배경으로 인한 OCR의 어려움은 실사 데이터 세트 학습을 통해 완화될 수 있을 것으로 기대된다.

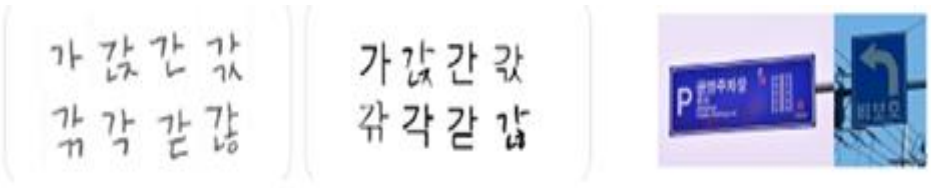

그림 3.4 학습 데이터 세트 예시 (손글씨, 인쇄체, 실사 데이터)[15]

3.3.4 학습 데이터 세트 설정

실험 1에서는 위에서 언급한 세 가지의 데이터 세트를 가지고 조합할 수 있는 모든 경우의 수의 데이터 세트를 바탕으로 모델들을 각각 학습시킨 후 성능 비교를 진행하였다. (2*2*2 - 1 = 7가지)

3.4 CNN 모델 조합

기존 연구에서 CNN 모델로 VGG, ResNet 모델을 사용했으나 이후 발전된 CNN 모델이 등장하였다. 한글에 비해 복잡도가 낮은 영어의 경우 기존의 모델로도 충분히 성능이 뛰어났으나 복잡도가 높아 성능이 낮았던 한글의 경우 더 발전된 CNN 모델을 이용하여 Feature Extraction을 진행할 시에 더 높은 성능이 기대되어 실험 2에서는 추가로 7개의 CNN 모델을 구축하여 실험을 진행하였다. 본 절에서는 7개의 모델에 대한 자세한 설명과 더불어 구조에 대해서 서술한다.

3.4.1 ResNext

ResNext 모델은 ResNet 구조의 네트워크에 Inception 모델에서 새롭게 사용한 "Cardinality" 개념을 도입한 모델이다[16]. ResNext 모델은 입력 Feature를 쪼개서 Convolution 연산을 각각 진행하여 서로 다른 Weight를 구한 뒤 합쳐주는 Split-Transform-Merge 연산을 진행하며, 이때 쪼개는 경로의 수를 "Cardinality" 라고 정의한다. 그림 3.5의 왼쪽 ResNet 구조와 같이 진행되는 연산을 오른쪽과 같이 입력 Feature를 쪼개서 연산을 진행한 후, 합치는 연산을 진행한다. 기본적인 CNN 성능에서 ResNet보다 높은 성능을 보였으며 본 논문에서는 기본적인 ResNext 모델과 변형을 진행한 ResNextv2, ResNextv3 모델을 실험하였다. 모델 구조는 다음 표4.1, 4.2, 4.3과 같다.

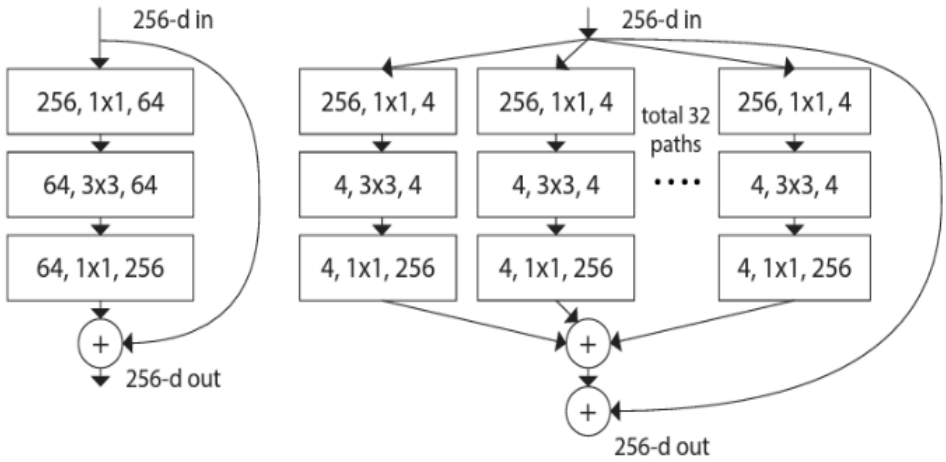

그림 3.5 ResNet 구조와 ResNext 구조 비교[16]

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 32 k: 3 * 3	100 * 32
Conv2	c: 64 k: 3 * 3	100 * 32
Pool1	k: 2 * 2 s: 2 * 2	50 * 16
Block1	c: 128, k: 1 * 1 c: 128, k: 3 * 3, C = 32 c: 128, k: 1 * 1 * 1	50 * 16
Conv3	c: 128 k: 3 * 3	50 * 16
Pool2	k: 2 * 2 s: 2 * 2	25 * 8
Block2	c: 256, k: 1 * 1 c: 256, k: 3 * 3, C = 32 c: 256, k: 1 * 1 * 2	25 * 8
Conv4	c: 256 k: 3 * 3	25 * 8
Pool3	k: 2 * 2 s: 1 * 2 p: 1 * 0	26 * 4
Block3	c: 512, k: 1 * 1 c: 512, k: 3 * 3, C = 32 c: 512, k: 1 * 1 * 5	26 * 4
Conv5	c: 512 k: 3 * 3	26 * 4
Block4	c: 512, k: 1 * 1 c: 512, k: 3 * 3, C = 32 c: 512, k: 1 * 1 * 3	26 * 4
Conv6	c: 512 k: 2 * 2 s: 1 * 2 p: 1 * 0	27 * 2
Conv7	c: 512 k: 2 * 2 s: 1 * 1 p: 0 * 0	26 * 1

표 3.3 ResNext 모델 구조 (C = Cardinality)

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 64 k: 3 * 3	100 * 32
Block1	c: 256, k: 1 * 1, s = 2 c: 256, k: 3 * 3, C = 32 c: 256, k: 1 * 1 * 3	50 * 16
Block2	c: 512, k: 1 * 1, s = 2 c: 512, k: 3 * 3, C = 32 c: 512, k: 1 * 1 * 4	25 * 8
Conv2	c: 512 k: 3 * 3 s: 1 * 2	26 * 4
Conv3	c: 512 k: 2 * 2 s: 1 * 2	27 * 2
Conv4	c: 512 k: 2 * 2 s: 1 * 2 p: 1 * 0	26 * 1
Conv5	c: 512 k: 3 * 3	26 * 1

표 3.4 ResNextv2 모델 구조 (C = Cardinality)

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 64 k: 3 * 3	100 * 32
Pool1	k: 2 * 2 s: 2 * 2	50 * 16
Block1	c: 256, k: 1 * 1, s = 2 c: 256, k: 3 * 3, C = 32 c: 256, k: 1 * 1 * 3	25 * 8
Conv2	c: 512 k: 3 * 3 s: 1 * 2	26 * 4
Conv3	c: 512 k: 2 * 2 s: 1 * 2	27 * 2
Conv4	c: 512 k: 2 * 2 s: 1 * 2 p: 1 * 0	26 * 1
Conv5	c: 512 k: 3 * 3	26 * 1

표 3.5 ResNextv3 모델 구조 (C = Cardinality)

3.4.2 Inception-ResNet

Inception-ResNet 모델은 Inception 구조의 네트워크에 ResNet 모델에서 새롭게 사용한 “Residual Connection” 개념을 도입한 모델이다[17]. 그림 3.6과 기존 Inception 모델의 구조인 오른쪽 모델에 Residual Connection을 도입하여 구성한 모델이며, ResNet보다 높은 성능을 보였다. 본 논문에서는 기본적인 Inception-ResNet 모델과 변형을 진행한 Inception-ResNetv2, Inception-ResNetv3, Inception-ResNetv4 모델을 실험하였다. 모델 구조는 다음 표 3.6, 3.7, 3.8, 3.9과 같다.

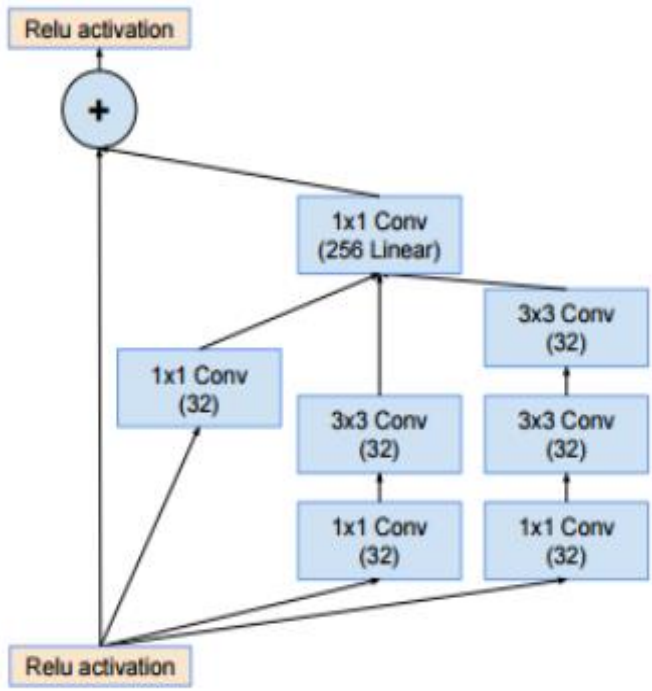

그림 3.6 Inception-ResNet 블록 구조[17]

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 32 k: 3 * 3	100 * 32
Conv2	c: 64 k: 3 * 3	100 * 32
Pool1	k: 2 * 2 s: 2 * 2	50 * 16
Conv3	c: 128 k: 3 * 3	50 * 16
Pool2	k: 2 * 2 s: 2 * 2	25 * 8
Branch1	c: 48 k: 1 * 1 c: 32 k: 5 * 5	25 * 8
Branch2	c: 64 k: 1 * 1 c: 32 k: 3 * 3 c: 32 k: 3 * 3	25 * 8
Branch3	c: 64 k: 1 * 1	25 * 8
Concat	Concat(Branch1, Branch2, Branch3)	25 * 8
Block1	c: 32, k: 1 * 1 c: 32, k: 3 * 3 -> c : 192 c: 64, k: 1 * 1 * 5	25 * 8
ReductionA	c: 448 s: 1 * 2 p: 1 * 0	26 * 4
Block2	c: 384, k: 1 * 1 c: 256, k: 3 * 3 -> c : 448 c: 512, k: 1 * 1 * 5	26 * 4
ReductionB	c: 1152 s: 1 * 2 p: 1 * 0	27 * 2
Conv4	c: 1152 k: 2 * 2 s: 1 * 1 p: 0 * 0	26 * 1

표 3.6 Inception-ResNet 모델 구조

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 32 k: 3 * 3	100 * 32
Conv2	c: 64 k: 3 * 3	100 * 32
Pool1	k: 2 * 2 s: 2 * 2	50 * 16
Conv3	c: 128 k: 3 * 3	50 * 16
Pool2	k: 2 * 2 s: 2 * 2	25 * 8
Branch1	c: 48 k: 1 * 1 c: 32 k: 5 * 5	25 * 8
Branch2	c: 64 k: 1 * 1 c: 32 k: 3 * 3 c: 32 k: 3 * 3	25 * 8
Branch3	c: 64 k: 1 * 1	25 * 8
Concat	Concat(Branch1, Branch2, Branch3)	25 * 8
Block1	c: 32, k: 1 * 1 c: 32, k: 3 * 3 -> c : 192 c: 64, k: 1 * 1 * 5	25 * 8
ReductionA	c: 448 s: 1 * 2 p: 1 * 0	26 * 4
Conv4	c: 448 k: 2 * 2 s: 1 * 2 p: 1 * 0	27 * 2
Conv4	c: 512 k: 2 * 2 s: 1 * 1 p: 0 * 0	26 * 1

표 3.7 Inception-ResNetv2 모델 구조

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 32 k: 3 * 3	100 * 32
Conv2	c: 64 k: 3 * 3	100 * 32
Pool1	k: 2 * 2 s: 2 * 2	50 * 16
Conv3	c: 128 k: 3 * 3	50 * 16
Pool2	k: 2 * 2 s: 2 * 2	25 * 8
Branch1	c: 48 k: 1 * 1 c: 32 k: 5 * 5	25 * 8
Branch2	c: 64 k: 1 * 1 c: 32 k: 3 * 3 c: 32 k: 3 * 3	25 * 8
Branch3	c: 64 k: 1 * 1	25 * 8
Concat	Concat(Branch1, Branch2, Branch3)	25 * 8
Block1	c: 32, k: 1 * 1 c: 32, k: 3 * 3 -> c : 192 c: 64, k: 1 * 1 * 5	25 * 8
ReductionA	c: 448 s: 1 * 2 p: 1 * 0	26 * 4
Block2	c: 384, k: 1 * 1 c: 256, k: 3 * 3 -> c : 448 c: 512, k: 1 * 1 * 5	26 * 4
ReductionB	c: 1152 s: 1 * 2 p: 1 * 0	27 * 2
Block3	c: 192, k: 1 * 1 c: 256, k: 3 * 3 -> c : 1152 * 5	27 * 2
Conv4	c: 512 k: 2 * 2 s: 1 * 1 p: 0 * 0	26 * 1

표 3.8 Inception-ResNetv3 모델 구조

Layers	Configurations	Output
Input	Image	100 * 32
Conv1	c: 32 k: 3 * 3	100 * 32
Conv2	c: 64 k: 3 * 3	100 * 32
Pool1	k: 2 * 2 s: 2 * 2	50 * 16
Conv3	c: 128 k: 3 * 3	50 * 16
Pool2	k: 2 * 2 s: 2 * 2	25 * 8
Branch1	c: 48 k: 1 * 1 c: 32 k: 5 * 5	25 * 8
Branch2	c: 64 k: 1 * 1 c: 32 k: 3 * 3 c: 32 k: 3 * 3	25 * 8
Branch3	c: 64 k: 1 * 1	25 * 8
Concat	Concat(Branch1, Branch2, Branch3)	25 * 8
Block1	c: 32, k: 1 * 1 c: 32, k: 3 * 3 -> c : 192 c: 64, k: 1 * 1 * 10	25 * 8
ReductionA	c: 448 s: 1 * 2 p: 1 * 0	26 * 4
Block2	c: 384, k: 1 * 1 c: 256, k: 3 * 3 -> c : 448 c: 512, k: 1 * 1 * 20	26 * 4
ReductionB	c: 1152 s: 1 * 2 p: 1 * 0	27 * 2
Block3	c: 192, k: 1 * 1 c: 256, k: 3 * 3 -> c : 1152 * 9	27 * 2
Conv4	c: 512 k: 2 * 2 s: 1 * 1 p: 0 * 0	26 * 1

표 3.9 Inception-ResNetv4 모델 구조

3.5 데이터 후처리

본 연구에서는 OCR 모델이 인식한 결과에 후처리를 적용하여 성능을 올리고자 하였다. 본 절에서는 후처리에 대한 자세한 설명과 더불어 후처리를 적용한 동기에 대해서 서술한다.

3.5.1 글자의 색 변환

학습에 사용된 데이터 중 손글씨와 인쇄체 데이터의 경우 흰 바탕에 검은색 글씨로 이루어져 있다. 그러나 그림 3.7와 같이 대부분의 동영상 이미지의 경우 검은색 글씨가 아닌 흰색 글씨, 혹은 흰색과 같이 밝은 계열의 노란색, 연두색 등과 같은 색깔로 쓰여 있는 경우가 많았으며 이러한 이유로 손글씨와 인쇄체 데이터만으로 학습한 모델의 경우 테스트 성능이 매우 낮았다. 이러한 문제를 해결하기 위해 테스트 진행 시 원본 이미지와 색깔을 반전시킨 이미지에 대해 동시에 글자 인식을 진행하여 모델이 출력하는 Confidence Score, 즉 결과의 확신도를 나타내는 지수가 높은 인식 결과를 선택하도록 하였다.

그림 3.7 테스트 데이터 예시(1)

3.5.2 맞춤법 교정

특정 이미지는 사람이 보기에도 글자가 헷갈리는 경우가 있다. 예를 들어 그림 3.8의 “청바지엔” 단어의 경우 이미지로만 보았을 때, “척바지엔” 으로 인식될 수 있으나 사람은 맞춤법을 바탕으로 하여 “척바지엔” 이라는 단어가 아닌 “청바지엔” 단어로 인식한다. 그러나 컴퓨터의 경우 맞춤법과 같은 사전 지식은 학습되지 않았으므로 인식된 결과에 맞춤법 교정을 진행하여 성능을 향상시키고자 하였다. 맞춤법 교정 모델의 경우 파이썬 오픈 라이브러리 py-hanspell을 이용하였다.

그림 3.8 테스트 데이터 예시(2)

제 4 장 실험 결과

4.1 실험 환경

본 연구에서는 이미지 데이터의 학습 효율성을 높이기 위해 입력 이미지 크기는 100 * 32 크기로 고정하였으며, RGB 값을 모두 흑백으로 전환하는 GrayScale 기법을 도입하여 학습 시간을 단축하였다. 학습 데이터로 AIHub에서 제공한 한국어 글자체 이미지를 사용하였으며 각 데이터의 수는 손글씨 데이터는 562,242개, 인쇄체 데이터는 1,707,107개, 실사 데이터는 1,149,675개다.

4.2 평가 및 분석

본 연구에서는 테스트 이미지 데이터 세트를 동영상으로부터 구축하였으며 3가지 평가지표를 사용한다. 본 절에서는 테스트 이미지 세트, 평가지표 및 실험 결과에 관해서 서술한다.

4.2.1 테스트 데이터 세트

본 연구에서는 테스트 이미지 데이터 세트로 유튜브 동영상을 사용하였다. 자막이 존재하는 동영상을 선별하여 일정 프레임 단위에 맞춰 캡처를 진행하여 동영상을 이미지의 묶음으로 변환한다. 이후에 Text Detection 모델인 “CRAFT” 를 활용하여

이미지 내에서 텍스트가 존재하는 부분의 좌표를 얻는다[18]. 얻은 좌표를 바탕으로 캡처된 이미지에서 텍스트만 있는 부분을 잘라낸 뒤에 텍스트를 입력하여 테스트 데이터 세트를 구축하여 사용하였다.

4.2.2 평가 지표

기본적인 평가 방법으로는 Word Recognition Accuracy (WRA) 지수와 Levenshtein 거리를 사용했다. WRA는 정답 단어와 예측된 단어를 비교하여 정확하게 일치하는지를 판단하는 방법이다. 수식은 다음과 같다.

$$\text{WRA} = \frac{Number\ of\ Correctly\ Recognized\ Words}{Total\ Number\ of\ Words} * 100 \qquad (1)$$

그러나 WRA의 경우 예측이 틀린 단어에 대해서 정답과의 유사도를 측정할 수가 없다는 단점이 있다. 따라서 이를 보완하고자 Levenshtein 거리 또한 사용하였다. Levenshtein 거리는 정답 단어와 예측 단어 사이의 편집 거리를 측정하는 방법이다. 편집 거리는 예측 단어를 정답 단어로 바꾸기 위해 몇 번의 연산(삽입, 대체, 대체)을 실행해야 하는지를 의미한다. 수식은 다음과 같다.

$$L(a,b) = \begin{cases} |a| & if\ |b| == 0 \\ |b| & if\ |a| == 0 \\ L\big(tail(a), tail(b)\big) & if\ a[0] == b[0] \\ 1 + \min \begin{cases} L(tail(a), b) \\ L(a, tail(b) & otherwise \\ L(tail(a), tail(b) \end{cases} \end{cases} \quad (2)$$

한글의 글자는 초성, 중성, 종성으로 이루어져 있다. 따라서 위의 방법을 글자 단위가 아닌 초성, 중성, 종성 단위의 Levenshtein 거리인 Jamo Levenshtein 거리 또한 사용하여 평가하였다.

$$\begin{gathered} \text{if s1} == \text{가감}, \text{s2} == \text{가김} \\ \mathrm{L(s1, s2)} = 1, L_{jamo}(s1, s2) = \frac{1}{3} \end{gathered} \quad (3)$$

4.2.3 실험 결과

첫 번째 실험은 AIHub에서 공개한 3개의 데이터 (손글씨(H), 인쇄체(P), 실사 데이터(W))를 조합하여 만든 7개의 모델을 바탕으로 진행하였으며 모든 후처리 또한 진행하였다. 기본적인 실험 모델은 기존 연구에서 가장 뛰어난 성능을 보인 모델인 (ResNet −BiLSTM) 조합을 사용하였다. 총 7개의 모델과 함께 성능이 가장 높은 한글 OCR 오픈소스 라이브러리인 EasyOcr까지 실험을 진행하였다. 각 모델의 이름은 학습한 데이터의 종류를 의미한다.

model	WRA	LEV	JAMO
H	27.2704	60.3328	45.2392
P	61.1124	26.6236	20.0688
W	87.9084	7.236	4.6706
HP	58.0062	29.1482	21.5168
HW	88.8947	6.6457	4.4508
PW	87.6534	7.6298	5.0572
HPW	87.9776	7.6896	4.8946
EasyOCR	60.231	14.7282	7.9782

표 4.1 실험 1 모델 성능 비교

실사 데이터(W)를 학습 데이터로 사용하지 않은 (H, P, HP) 모델의 경우 낮은 성능을 보였으며, 인쇄체와 실사 데이터를 조합하여 학습한 모델(HW)이 가장 성능이 뛰어났다. 정확도를 나타내는 지수 WRA가 가장 높았으며, 정답과의 근접성을 의미하는 편집거리 Levenshtein, Jamo Levenshtein 거리 또한 가장 낮았다. 실사 데이터를 함께 학습하였을 때 모두 시중에 공개된 오픈 소스 라이브러리 EasyOCR보다 매우 좋은 성능을 보였으며, 인쇄체 데이터만을 학습한 모델(P)과 손글씨 데이터와 인쇄체 데이터를 학습한 모델(HP)의 경우 EasyOCR과 유사한 성능을 보였다.

두 번째 실험은 위에서 언급한 CNN 모델들을 조합하여 구성한 모델들을 바탕으로 진행하였다. 데이터는 첫 번째 실험에서 가장 뛰어난 조합이었던 HW 조합을 사용하였고, RNN 모델로 BiLSTM을 조합하였으며 데이터 후처리 또한 진행하였다.

model	WRA	LEV	JAMO
ResNet	88.8947	6.6457	4.4508
ResNext	90.4962	5.7823	3.8665
ResNextv2	85.9806	8.2505	5.3986
ResNextv3	89.9186	6.0003	3.9624
Inception-ResNet	90.0761	5.948	3.8723
Inception-ResNetv2	89.8924	5.887	3.9305
Inception-ResNetv3	89.6298	6.262	4.1252
Inception-ResNetv4	89.9974	5.948	3.8432

표 4.2 실험 2 모델 성능 비교

정확도를 나타내는 지수 WRA 및 글자 단위 편집 거리 Levenshtein 거리 측면에서 RexNext 모델이, 음절 단위 편집 거리 Jamo Levenshtein 거리 측면에서 Inception-ResNetv4 모델이 가장 성능이 뛰어났다.

4.2.4 실패 사례 분석

인식을 실패한 경우는 크게 두 가지 유형으로 나눌 수 있다. 첫 번째 유형은 글자가 하나 더 있다고 인식하는 경우였다. 이러한 유형의 예시 들을 살펴본 결과, 표 4.3의 Type 1의 두 가지 이미지와 같이 한글과 함께 학습하지 않은 영어 혹은 이모티콘이 들어가 있을 때 학습하지 않은 문자를 가장 유사한 단어로 변형시켜 결과를 도출하였으며 그 예시로 “#탈출” 의 경우 “파탈출” 로, “#성공적” 의 경우 “부성공적” 으로 인식하였다. 이는 동영상에서 쓰이는 다양한 이모티콘과 영어를 함께 학습시킨다면 개선될 수 있을 것으로 기대된다. 두 번째 유형은 맞춤법을 헷갈리는 경우였다. 이러한 유형의 케이스들을 살펴본 결과 시각적으로 유사하게 보이는 (ㅁ – ㅇ) 과 같은 형태소 조합의 경우에서 실패를 겪는 경우가 많은 것으로 확인되었다. 표 4.3의 Type 2의 두 가지 이미지와 같이 “테일러” 의 경우 “테밀러” 로, “응용” 의 경우 “몽용” 으로 인식하였다. 이러한 문제는 맞춤법 교정 후처리를 통해 해결하려고 시도하였으나 “테일러” 의 경우 사전에 등재되지 않은 외국어이므로 해결하지 못한 것으로 보이며 “몽용” 의 경우 사전에 이미 존재하는 단어이기에 “응용” 으로 고치지 않은 것으로 보인다. 모델의 인식 결과에 각 단어들이 사용되는 빈도를 모델의 인식 결과에 반영하여 맞춤법 교정을 진행한다면 개선될 것으로 기대된다.

	Type 1	Type 2
Image		
Index	탈출	응용
Predict	파탈출	몽용
Image		
Index	성공적	테일러
Predict	부성공적	테밀러

표 4.3 인식 실패 예시

제 5 장 결론

본 논문에서는 기존 OCR 모델들이 갖고 있던, 한글 텍스트 성능이 매우 떨어지는 문제점을 개선하고 동영상 데이터 분석에 최적화된 OCR 모델을 제안했다. 딥러닝 기법 중 하나인 CRNN 을 활용하여 모델을 구현하였으며 다양한 동영상을 테스트 데이터 세트로 추출하여 해당 모델을 검증하였다. 본 연구에서는 최근 공개된 한국어 이미지 데이터 학습을 통해 한글 인식이 충분히 가능함을 제시하였으며 기존에 가장 성능이 좋았던 오픈소스 라이브러리 EasyOCR 보다 성능이 매우 뛰어난 모델을 제시하였다.

그러나, 제안된 OCR 모델은 다음과 같은 문제점을 지닌다. 첫째는 모델이 학습하지 않은 문자로 인한 인해 발생하는 문제이다. 현재 논문에서 제안한 OCR 모델은 한글의 글자들만을 학습하였으며 동영상에서 자주 쓰이는 영어, 이모티콘의 경우 학습되지 않았다. 따라서 텍스트 인식을 시도할 때, 영어 혹은 이모티콘이 등장하는 경우 시각적으로 가장 유사한 한글의 글자로 인식하는 문제가 발생하였다. 이를 해결하기 위해 추가적인 영어, 이모티콘 데이터를 확보하여 학습을 진행한다면 더 좋은 모델이 제안될 수 있을 것으로 기대된다.

두번째는 시각적으로 유사한 형태소로 인해 발생하는 문제이다. 이는 다양한 글자의 종류 및 글씨체가 존재하는 한글의 특성으로 인해 발생하는 문제이다. "ㅁ - ㅇ" 혹은 "ㅅ - ㅈ" 같이 유사한 형태소가 존재하며 이는 쓰이는 글자체에 따라서 모델의 인식 성능에 부정적인 영향을 끼치게 된다. 이를 해결하고자 맞춤법 교정을

통해 인간의 사전지식을 반영하고자 하였으나 맞춤법 교정 성능의 부족으로 인해 해결이 안되는 경우가 다수 존재하였다. 따라서 맞춤법 교정을 진행하는 과정에서 글자가 사용되는 빈도를 반영하여 더욱 잘 쓰이는 단어들로 교정되도록 문제점을 개선한다면 더 좋은 모델이 제안될 수 있을 것으로 기대된다.

본 연구에서는 한국어 텍스트가 나타나 있는 이미지로 구성된 동영상의 텍스트 인식 성능을 향상시켰다. 향후 연구로 자연어 처리 모델을 인식된 텍스트에 적용하고자 한다. 앞서 서론에서 언급한 대로 브랜드의 유행을 브랜드의 노출 빈도를 통해 분석하고자 했다면 나아가 인식된 텍스트 문장에 대표적인 자연어 처리 모델 중 하나인 감성 분석 모델을 적용하여 브랜드에 대한 긍정, 부정 특성과 함께 브랜드의 특징까지 분석할 수 있을 것으로 기대된다. 또한, 다양한 분야의 동영상의 자막을 본 논문에서 제안한 모델로 추출한 뒤에 자연어 처리 모델을 활용하여 텍스트를 요약하면 자동으로 동영상의 내용을 요약하는 모델이 제시될 수 있을 것으로 기대된다.

참고 문헌

[1] 디렉터 짱구대디, 2022. 예전과 많이 달라진 국내 브랜드 22 FW 룩북 탐방 2편, https://www.youtube.com/watch?v=_Px16tgkNeM.

[2] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521.7553 (2015): 436-444.

[3] Jaderberg, Max, Karen Simonyan, and Andrew Zisserman. "Spatial transformer networks." Advances in neural information processing systems 28 (2015).

[4] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems 27 (2014).

[5] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

[6] Shi, Baoguang, Xiang Bai, and Cong Yao. "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition." IEEE transactions on pattern analysis and machine intelligence 39.11 (2016): 2298-2304.

[7] Liu, Wei, Chaofeng Chen, and Kwan-Yee Wong. "Char-net: A character-aware neural network for distorted scene text recognition." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 32. No. 1. 2018.

[8] Lee, Chen-Yu, and Simon Osindero. "Recursive recurrent nets with attention modeling for ocr in the wild." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[9] Baek, Jeonghun, et al. "What is wrong with scene text recognition model comparisons? dataset and model analysis." Proceedings of the IEEE/CVF international conference on computer vision. 2019.

[10] Liu, Wei, et al. "Star-net: a spatial attention residue network for scene text recognition." BMVC. Vol. 2. 2016.

[11] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[12] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Communications of the ACM 60.6 (2017): 84-90.

[13] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[14] 한국표준협회, 2019. KS X 1001, https://standard.go.kr/KSCI/standardIntro/getStandardSearchView.do?ksNo=KSX1001.

[15] 키니앤파트너스, 2019. 한국어 글자체 이미지. https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=81.

[16] Xie, Saining, et al. "Aggregated residual transformations for deep neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[17] Szegedy, Christian, et al. "Inception-v4, inception-resnet and the impact of residual connections on learning." Thirty-first AAAI conference on artificial intelligence. 2017.

[18] Baek, Youngmin, et al. "Character region awareness for text detection." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

Abstract

Optical Korean Character Recognition using CRNN with BiLSTM for video analysis

Eojin Kim
Department of Industrial Engineering
The Graduate School
Seoul National University

Deep learning-based optical character recognition (OCR) has excellent performance in English and numbers. However, due to the characteristics of Hangeul, which has many types of characters, it shows low performance in Korean recognition. In addition, most models show high performance for images with the same background, but low performance for images with various backgrounds. In this study, we attempted to improve the Korean recognition performance by compressing the Korean character type, combining training data, changing the model structure, and post-processing the results. Currently, the results are much better than the results of EasyOCR, the best Korean OCR open source library. Through this study, the performance of the model's Korean recognition was improved. In addition, a small number of failure cases are analyzed and presented for future research directions.

Keywords: OCR, Korean Recognition, Video Analysis, CRNN, BiLSTM
Student Number: 2021-2811

감사의 글

석사과정 재학 기간동안 저의 연구를 지지해주신 조성준 교수님께 감사의 말씀을 드립니다. 함께 데이터마이닝과 인공지능 분야에 대한 다양한 연구를 진행하고 도움을 주셨던 서울대학교 산업공학과 데이터마이닝 연구실 모든 식구들께 감사드립니다.