딥러닝을 활용한 한글문장 OCR연구

박선우

모두의연구소
psw261@gmail.com

A Study on the OCR of Korean Sentence Using DeepLearning

Sun-Woo Park
Modulabs

요 약

한글 OCR 성능을 높이기 위해 딥러닝 모델을 활용하여 문자인식 부분을 개선하고자 하였다. 본 논문에서는 폰트와 사전데이터를 사용해 딥러닝 모델 학습을 위한 한글 문장 이미지 데이터를 직접 생성해보고 이를 활용해서 한글 문장의 OCR 성능을 높일 다양한 모델 조합들에 대한 실험을 진행했다. 딥러닝 모델은 STR(Scene Text Recognition) 구조를 사용해 변환, 추출, 시퀀스, 예측 모듈 각 24가지 모델 조합을 구성했다. 딥러닝 모델을 활용한 OCR 실험 결과 한글 문장에 적합한 모델조합은 변환 모듈을 사용하고 시퀀스와 예측 모듈에는 BiLSTM과 어텐션을 사용한 모델조합이 다른 모델 조합에 비해 높은 성능을 보였다. 해당 논문에서는 이전 한글 OCR 연구와 비교해 적용 범위를 글자 단위에서 문장 단위로 확장하였고 실제 문서 이미지에서 자주 발견되는 유형의 데이터를 사용해 애플리케이션 적용 가능성을 높이고자 한 부분에 의의가 있다.

주제어: OCR, 문자인식, 문자탐지, 딥러닝

1. 서론

광학 문자 인식 OCR(Optical Character Recognition)은 사람이 쓰거나 기계로 인쇄한 문자의 영상을 이미지 스캐너로 획득하여 기계가 읽을 수 있는 문자로 변환하는 것이다. 회사 및 관공서의 계약서 및 서류 문서들은 지금까지도 인쇄물로서 보관되는 곳이 많다. 이러한 정보자산들의 디지털화에 있어서 가장 필요하고도 중요한 기술이 바로 OCR이다. 그러나 오픈소스를 포함한 상용 OCR 엔진은 영문에 초점이 맞춰져 있어 한글 문서에 적용할 수 있는 기술연구가 필요하다.

OCR 기술은 크게 문자탐지(text detection)와 문자인식(text recognition)으로 구성되어 있고 OCR 기술이 어려운 점은 크게 3가지로 정리된다. 첫째, 문서 이미지 속 글자들은 정형화되어 있지만, 손글씨 및 서명은 비정형화되어 있어 분별이 어렵다. 둘째, 배경이 복잡한 경우 배경과 문자의 구분이 힘들다. 그리고 다양한 간섭요소, 예를 들면 노이즈, 왜곡, 글자 사이 밀도, 저해상도로 인해 식별에 어려움이 있다. [1] 한글에 대한 OCR이 더 어려운 이유는 영문과 비교해 분류해야 할 글자 수가 실험기준으로 (26 → 900) 35배 정도 차이가 나 정확도를 높이는 데 어려움이 있다. 이전까지의 한글 OCR 연구는 글자 단위에 초점이 맞춰져 있었다. 또한 사용된 모델구조 역시 CNN(Convolutional Neural Network)이나 RNN(Recurrent Neural Network)과 같이 단일 모델로만 구성되어 있어 성능을 높이는 데 한계가 있었다. 기존 문자탐지 모델의 스코어 맵과 어피니티 맵을 분석해 본 결과 문자가 존재할 영역을 글자 단위별로 정확히 인식하였고 글자사이 여백에 대한 분류를 통해 단어 간 분류까지 가능했다. 따라서 OCR 성능을 높이기 위해서 문자인식 부분에 초점을 맞추어 연구 방향을 설정하였다.

본 논문에서는 기존 영문 문자인식 모델조합을 찾는 데 사용된 4단계 STR(Scene Text Recognition) 모델구조(변환, 추출, 시퀀스, 예측)를 사용해 실험을 진행하였다. 또한 실험을 위한 데이터 세트를 한국어 학습용 어휘목록과 한글 폰트를 사용해 직접 생성하여 진행하였다.

논문의 구성은 OCR 관련 이전연구에 대해 모듈별로 주요특성을 살펴보고 한글 문자인식 모델조합을 찾는 딥러닝 모델 구성, 실험구현 상세과정에 관해 설명한다. 마지막으로 실험 결과에 대한 분석과 결과를 정리했다.

2. 관련 연구

2.1 문자탐지(text detection)

기존의 문자탐지는 글자/단어 후보생성→ 후보 필터링 → 그룹화와 같은 여러 단계로 나누어져 있었다. 따라서 모델 학습 중의 튜닝이 어려웠고, 완성된 모델의 속도도 느려서 실시간 탐지에 적용하기 힘들었다. Textboxes [2]는 물체 감지에서 큰 성능 향상을 보여준 SSD [3] 논문을 본떠 만들었으며, 단일 네트워크로 구성되어 있기 때문에 기존의 모델들에 비해 현저히 빠른 성능과 정확도 향상을 보였다. 그러나 Textboxes는 문자와 비문자 그리고 경계 박스에 대한 회귀분석 방식이다 보니 각 이미지가 매우 가깝게 자리 잡고 있다면 문자와 비문자를 구별하기가 힘들다. 이를 개선하기 위해 나온 방법이 의미 분할 (semantic segmentation) 방법이다. 이 방법은 분할의 기본 단위를 클래스로 하여 같은 클래스에 해당하는 사물을 예측하고 마스크 상에 동일한 색상으로 표시하는 방법이다. PixelLink [4] 에서 사용한 인스턴스 분할 (instance segmentation)은 분할의 기본 단위를 사물로 하여, 동일한 클래스에 해당하더라도 서로 다른 사물에 해당하면 이들을 예측 마스크 상에 다른 색상으로 표시한다. PixelLink는 위치회귀(location regression) 없이 텍스트 박스에서 직접 분할결과를 추출해 더 적은 수용영역이 필요하고, 이러한 부분은 학습을 더 쉽게 만든다.

FOTS (Fast Oriented Text Spotting) [5]는 종단 간 접근방식을 적용해 탐지와 인식 모듈이 동시에 훈련됨으로써 상승한 인식 결과가 다시 탐지 모듈의 정확도를 높이는 효과를 가져온다.

2.2 문자인식(text recognition)

문자인식 부분에서는 특성을 추출하는 CNN과 시계열 모델인 RNN을 통합하여 하나의 통일된 네트워크 구조의 CRNN [6]이 제안되었다. CRNN은 먼저 CNN을 통해 입력 이미지로부터 특성 시퀀스를 추출하고 이 특성 시퀀스들을 RNN의 입력값으로 하여 이미지의 텍스트 시퀀스를 예측한다. 예측된 텍스트 시퀀스를 텍스트로 변환하여 결과를 출력한다. 이 모델은 미리 정해진 어휘에 제한되지 않고, 임의 길이의 시계열 데이터를 다룰 수 있는 특징이 있다.

GRCNN (Gated Recurrent Convolution Neural Network) [7]은 recurrent convolution 레이어에 있는 컨텍스트 모듈을 제어하기 위해 RCNN 에 게이트가 더해진 모델이다. 게이트(gate)는 컨텍스트 모듈 제어뿐만 아니라 CNN의 정보와 RNN의 정보를 균형 있게 제어하는 역할을 한다.

특성 영역과 타깃 사이의 정확한 alignment를 얻지 못하는 현상을 어텐션 드리프트(Attention drift)라고 한다. FAN (Focusing Attention Network) [8] 은 이러한 문제를 해결하기 위해 고안된 방법으로 2가지 요소로 구성된다. 먼저 어텐션 네트워크는 글자 타깃을 인식하는 데 사용되고 포커싱 네트워크는 어텐션 네트워크가 타깃 지역에 적절히 어텐션을 갖는지 측정해서 어텐션을 조절하는 역할을 한다.

조금은 다른 접근방식으로 문자인식 모델들의 비교에서 잘못된 부분을 분석한 논문도 있다. 해당 논문 [9] 에서는 훈련 데이터 세트와 평가 데이터 세트가 일치하지 않음으로써 성능에서 차이가 발생하는 것을 증명하고 STR(Scene Text Recognition)의 구조를 사용하여 모듈조합 간 성능 비교분석을 진행했다.

2.3 종단 간 문자인식(end-to-end text recognition)

Aster (Attentional Scene Text Recognizer with Flexible Rectification) [10] 는 교정 네트워크와 인식 네트워크를 종단 간으로 연결하는 모델이다. 교정 네트워크는 입력 이미지를 새로운 이미지로 적응적으로 변환하여 이미지 속 기울어지거나 왜곡된 텍스트를 교정한다. 인식 네트워크는 어텐션 기반의 seq2seq 모델로 변환된 이미지로부터 글자 시퀀스를 예측한다.

문자탐지와 문자인식은 서로 상이함으로 인해 최적화의 어려움이 있어 통합된 구조로 만들기 어려웠다. TextSpotter [11] 는 text-alignment 레이어와 캐릭터 어텐션 메커니즘을 사용하여 통합 구조를 만들었다. 문자탐지와 문자인식 두 개 모듈의 통합은 서로 컨볼루션 특성을 공유함으로써 상호보완적으로 작동하고 종단 간으로 훈련이 가능하여 더 높은 성능을 보인다.

Tesseract OCR [12] 엔진은 상업용 OCR 엔진의 성능을 개선하기 위해 1985년 HP사 연구소에서 시작된 OCR 오픈소스이다. 최근 릴리즈된 버전은 LSTM (Long Short-Term Memory models) 기반의 버전까지 출시되었으며 100개 이상의 언어를 지원한다. 하지만 한글 OCR 성능은 정확도가 낮아 실제 애플리케이션에 적용하기가 힘들다는 한계가 있다.

2.4 한글 OCR 이전연구

한글에 대한 OCR 이전연구로는 이미지 프로세싱을 통해 이미지에서 글자영역을 추출하고, 이를 학습 데이터로 활용한 딥러닝으로 한글 OCR의 정확도를 향상하는 방법을 제안한 연구가 있다. [13] 한글의 초성, 중성, 종성의 모든 경우의 수를 조합하여 글자의 이미지를 생성, 그 이미지를 CNN을 이용해 딥러닝으로 학습시킨다. 다른 연구로는 한글 필기체 인식 문제에서 펜촉의 움직임 정보를 기반으로 하는 온라인 방식의 해법 체계를 정리하고, RNN 기반의 딥러닝 기법의 가능성을 예비 실험을 통해 확인한 결과를 정리한 연구가 있다. [14] 조합 가능한 모든 한글 글자 이미지를 생성하거나 SERI95a, PE92 데이터 세트를 사용하였으나 글자 단위로만 진행되어 한계를 가진다.

3. 한글 문장 **OCR**을 위한 딥러닝 모델 구성

3.1 문자탐지 부분과 문자인식 부분의 비교

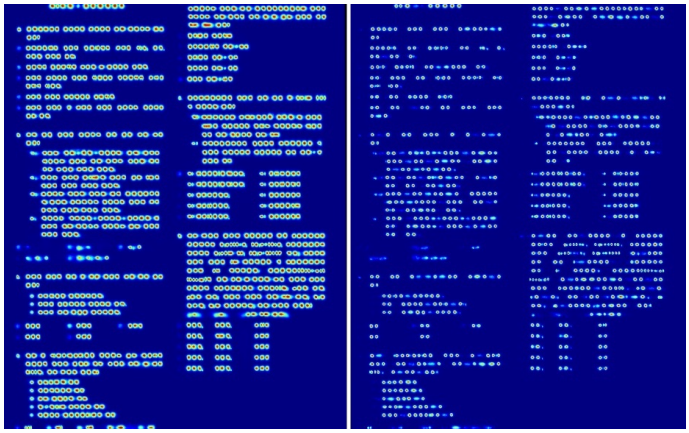

그림 1. 글자기반 문자탐지모델 스코어맵과 어피니티맵

그림 1은 글자 단위 문자탐지 모델을 사용했을 때 생성되는 글자 스코어 맵과(왼쪽) 글자 사이 여백을 표시하는 어피니티 맵(오른쪽) 이다. 그림 1에서 보는 것과 같이 문자 영역을 정확히 인식하였으며, 글자뿐만 아니라 글자 사이 간격까지 찾아내 하나의 단어는 동일한 바운딩 박스 안에 포함되는 모습을 그림 2에서 확인할 수 있다.

글자영역을 찾는 것보다 찾은 글자영역을 바탕으로 정확한 글자를 인식하는 것이 한글 문장의 OCR 성능을 높이는데 주요하다고 판단하여 문자인식 모델에 중점을 두어 이후 실험을 진행했다.

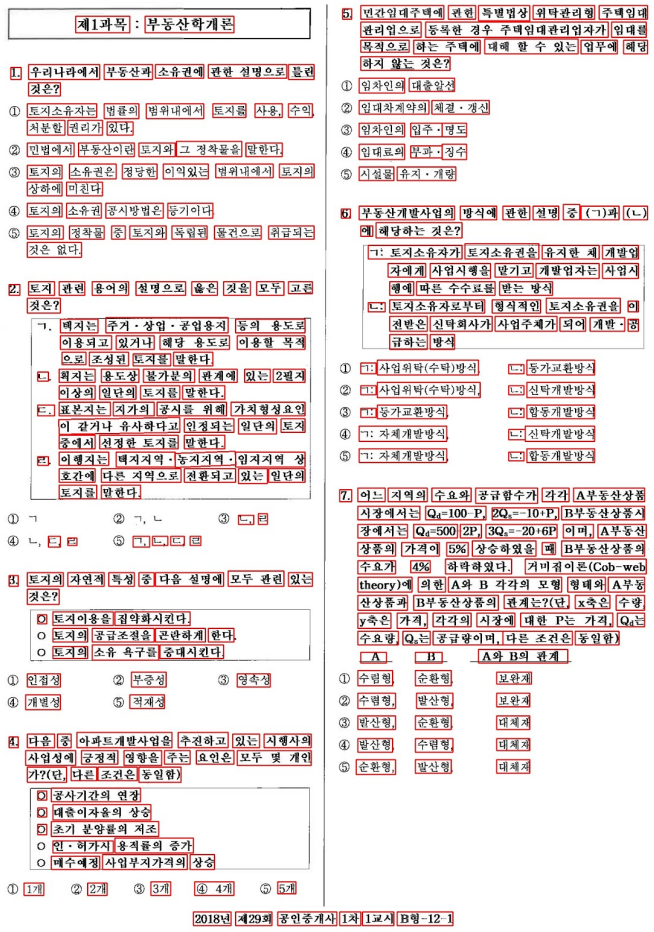

그림 2. 원본파일과 바운딩 박스

3.2 모델구조

기존 OCR 모델의 경우 크게 문자탐지와 문자인식으로 구성되어있는 반면, OCR을 위한 딥러닝 모델은 STR(Scene Text Recognition)의 구조를 가져와서 변환, 특성추출, 시퀀스, 예측 4개 모듈로 세분되어 구성되어있다.

먼저 변환 모듈은 TPS (Thin Plate Spline) 변환 사용 여부에 따라 모듈이 구분된다. TPS 변환은 데이터 보간 및 평활화를 위한 스플라인 기반 기술로 이미지 속 문자가 기울거나 왜곡되어있는 경우 바로잡아 표준화된 이미지로 만드는 역할을 한다. 특성추출 모듈은 이미지에서 시각적 특성을 추출하는 부분으로 VGGNet, RCNN, ResNet 모델을 각각 사용하였다. VGGNet은 간단한 구조와 단일 네트워크에서 좋은 성능을 보여준다는 이유로 많은 네트워크에서 응용되고 있다. RCNN은 이미지 분류를 수행하는 CNN과 이미지에서 물체가 존재할 영역을 제안해주는 region proposal 알고리즘을 연결하여 높은 성능의 물체탐지를 가능하게 하는 모델이다. ResNet은 망이 깊어지는 경우 발생하는 그래디언트 소멸/폭발 부작용을 해결하기 위해 레이어의 입력을 레이어의 출력에 바로 연결하게 하는 스킵 커넥션 (skip connection)을 사용한 모델이다. 시퀀스 모델부분은 BiLSTM 사용 여부에 따라 모듈이 구분된다. BiLSTM은 앞에서 뒤, 뒤에서 앞, 모두 고려하는 양방향(bidirectional) 네트워크를 통해 LSTM의 성능 개선을 가능하게 한다. 예측 모듈은 CTC(Connectionist Temporal Classification)와 어텐션으로 모듈이 구분된다. CTC는 학습데이터에 클래스 라벨만 순서대로 있고 각 클래스의 위치는 어디 있는지 모르는 분할되지 않은 시퀀스 데이터의 학습을 위해서 사용하는 알고리즘이다. 문장 길이가 길고 층이 깊으면, 인코더가 압축해야 할 정보가 많아 정보 손실이 일어나고, 디코더는 인코더가 압축한 정보를 초반 예측에만 사용하는 경향을 보인다. 이 때문에 인코더-디코더 사이에 병목현상이 발생하고 이에 디코더 예측 때 가장 의미 있는 인코더 입력에 주목하게 만드는 어텐션 메커니즘이 제안되었다.

앞서 언급된 4개 모듈조합은 전체 경우의 수가 2 x 3 x 2 x 2 = 24가지가 나와 한글 문장에 가장 적합한 모듈조합을 찾고자 모든조합에 대한 실험을 진행했다.

4. 실험

4.1 데이터 세트

한글 문장 OCR과 관련하여 공개된 데이터 세트가 없어 딥러닝 모델을 훈련하기 위해선 한글 문장 데이터 세트를 직접 생성할 필요가 있었다. 데이터 세트를 생성하는 과정은 크게 다음과 같다. 1) 단어 사전에서 랜덤으로 단어를 선정 2) 한글 폰트 파일과 배경 3종(가우시안 노이즈, 순백색, Quasi crystal) 준비 3) 기본, 배경, 기울기, 왜곡, 흐리게 한 5가지 종류의 한글 문장 데이터 세트 생성.

한글 폰트는 네이버 나눔 글꼴(23종) [15] 과 폰트 코리아(76종) 폰트 [16] 총 99종을 준비했다. 한글 단어 사전은 총 5,965개의 단어와 974개 글자를 포함한 국립국어원의 한국어 학습용 어휘목록 [17] 을 사용했다. 그리고 문서 파일에서 가능한 변형으로 (기울기, 왜곡, 흐리게 하기, 배경 포함)을 선택하였고, 한 개 문장 데이터가 포함하는 단어의 수는 10개로 설정 후 그림 3과 같은 문장 데이터를 생성했다. 훈련데이터는 기본과 기울기 데이터를 각각 9,000개 포함하고 검증데이터는 왜곡, 흐리게 하기, 배경 포함 데이터를 각각 3,000개로 맞춰 훈련데이터와 검증데이터 비율을 2:1로 구성했다. 테스트 데이터 세트는 직접 생성한 5가지 종류 (기본, 기울기, 왜곡, 흐리게 하기, 배경 포함) 의 데이터 세트 2,700개 문장과 최근 공개된 AI 오픈 이노베이션 허브의

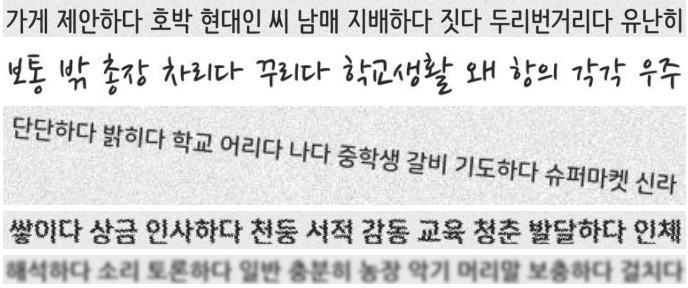

그림 3. 문자인식을 위한 생성 데이터. 위에서부터 차례로 기본,배경,기울기,왜곡,흐리게 한 문장 데이터

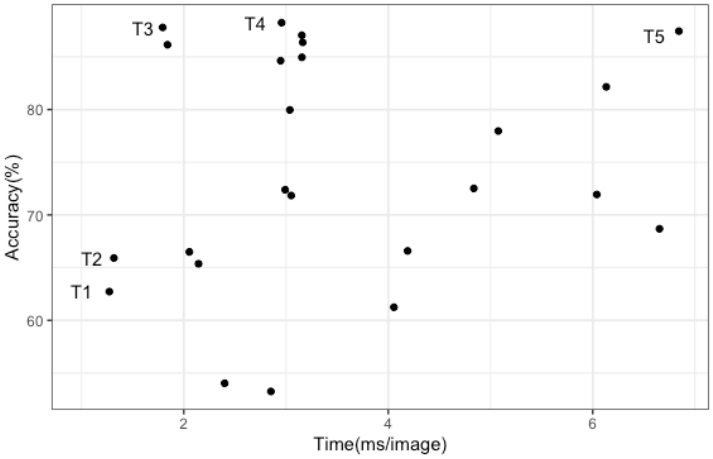

그림 4. 모델조합별 시간과 정확도 관계

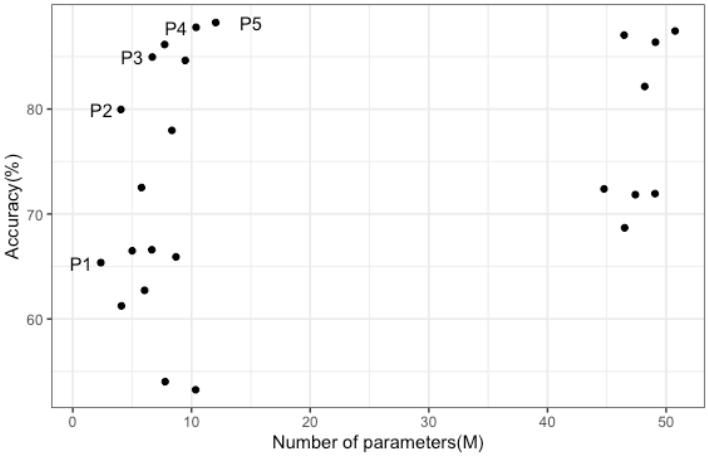

그림 5. 모델조합별 파라미터 수와 정확도 관계

표 1. 시간과 정확도 모델관계 프런티어 조합

#	변환	추출	시퀀스	예측	정확도 %	시간 ms	파라미터 $*10^6$
T1	None	VGG	None	CTC	62.72	1.27	6.04
T2	None	VGG	BiLSTM	CTC	65.91	1.31	8.69
T3	TPS	VGG	BiLSTM	CTC	87.79	1.79	10.38
T4	TPS	VGG	BiLSTM	Attn	88.24	2.95	12.04
T5	TPS	ResNet	BiLSTM	Attn	87.43	6.84	50.75

표 2. 파라미터 수와 정확도 모델관계 프런티어 조합

#	변환	추출	시퀀스	예측	정확도 %	시간 ms	파라미터 $*10^6$
P1	None	RCNN	None	CTC	65.36	2.14	2.35
P2	TPS	RCNN	None	CTC	79.96	3.03	4.05
P3	TPS	RCNN	BiLSTM	CTC	84.95	3.15	6.69
P4	TPS	VGG	BiLSTM	CTC	87.79	1.79	10.38
P5	TPS	VGG	BiLSTM	Attn	88.24	2.95	12.04

한국어 글자체 이미지 AI 데이터 [18] 중 인쇄체 현대 한글 6만 자, 단어 3만 자를 사용했다.

4.2 모델조합 비교

정확한 성능 측정을 위해 모델 조합별로 3번씩 훈련을 수행하였고 3번 결괏값의 평균값으로 정확도와 추론 시간, 파라미터 복잡도를 측정했다. 예측모델의 종류에 따라 예측모델이 CTC인 경우에는 CTC 로스를 어텐션인 경우에는 크로스엔트로피 로스를 사용해 정확도를 계산하였다. 정확도와 함께 Nltk 모듈에서 제공하는 edit_distance를 사용해서 두 문장 간의 유사도를 따로 표시했다.

실험 결과를 살펴보면 그림 4는 모델조합별 시간과 정확도의 관계를 그림 5는 파라미터 수와 정확도의 관계를 나타내고 있다. 표 1은 그림 4에서 경계의 끝에 위치해 좋은 성능을 보이는 5개 모델조합을 상세히 나타낸 표이다. 표 2는 그림 5에서 경계의 끝에 위치해 좋은 성능을 보이는 5개 모델조합을 상세히 나타낸 표이다.

가장 높은 정확도를 보인 모델조합은 TPS-VGG-BiLSTM -Attetion이다. 변환 모듈에서 TPS(Thin Plate Spline)를 사용할 때에 정확도는 최대 34%까지 향상되었으며, 파라미터 수의 증가량은 1.6(M)으로 일정하지만, 시간은 모델에 따라 증가하거나 감소하는 모습을 보였다. 추출 모듈에서 VGG와 ResNet이 RCNN보다 다소 정확도가 높게 나왔고 모듈간 파라미터 수를 비교했을 때 ResNet(47.7M), VGG(9.0M), RCNN(5.3M) 순으로 큰 값을 보였다. 시퀀스 모듈에서 BiLSTM 사용 여부는 다른 모듈조합과의 관계에 따라 정확도와 시간이 바뀌는 모습을 보였다. 시퀀스-예측 모듈에서 BiLSTM-어텐션의 조합이 BiLSTM 미사용-어텐션 조합보다 정확도를 높이는 효과를 보여 BiLSTM과 어텐션을 함께 사용했을 때 전체적인 정확도에 영향을 미쳤다.

4.3 실험환경

파이토치로 구현되어있는 문자인식 모델과 텐서플로우로 구현되어있는 문자인식 학습 데이터 생성 모델 소스를 베이스라인으로 구성하였다. 두 개의 오픈소스 모두 영문을 기반으로 작성되어있어 한글에 맞춰 동작할 수 있도록 변환 작업이 필요했다. 배치 사이즈는 192로, 반복 횟수는 300,000회, Adam Optimizer의 베타 값은 0.9, Adadelta의 decay rate는 0.95, eps는 1e-8 로 설정했다. 모델실험은 Tesla P40 * 2대와 GeForce GTX 1080 Ti * 4대로 3주간 진행하였다.

5. 결론

한글에 대한 OCR 연구는 그 중요성에 비해 다양한 연구가 진행되지 않았다. OCR 모듈별로 주요 특징을 정리해보고 한글 문자인식에 적합한 딥러닝 모델 조합을 찾는 실험을 진행했다. 실제 애플리케이션에 적용 가능성을 높이고자 한글 문장 OCR 데이터를 문서에서 발견되는 형태로 직접 생성하였다. 한글 문장에 적합한 모델 조합은 TPS-VGG-BiLSTM-Attetion이 다른 모델 조합에 비해 높은 정확도를 보였다. 정확도 향상으로 실제 서비스에 사용하려면 최근 주목받는 트랜스포머, 버트 모델을 고려해볼 만하다.

사 사

이 논문은 모두의연구소 Deep Learning College 과정과 정보통신산업진흥원의 2019년 고성능 컴퓨터 지원 사업의 지원을 받아 수행된 연구임

참고문헌

1. Yingying ZHU, Cong YAO, Xiang BAI, Scene Text Detection and Recognition: Recent Advances and Future Trends, pp. 1-2, 2015.
2. Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, Wenyu Liu, TextBoxes: A Fast Text Detector with a Single Deep Neural Network, pp. 1-3, 2016.
3. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, SSD: Single Shot MultiBox Detector, pp. 1-3, 2015.
4. Dan Deng, Haifeng Liu, Xuelong Li, Deng Cai, PixelLink: Detecting Scene Text via Instance Segmentation, pp. 1, 2018.
5. Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, Junjie Yan, FOTS: Fast Oriented Text Spotting with a Unified Network, 2018.
6. Baoguang Shi, Xiang Bai, Cong Yao, An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition, pp. 2-5, 2015.
7. Jianfeng Wang, Xiaolin Hu, Gated Recurrent Convolution Neural Network for OCR, pp. 3-5, 2017.
8. Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, Shuigeng Zhou, Focusing Attention: Towards Accurate Text Recognition in Natural Images, pp. 1-5, 2017.
9. Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, Hwalsuk Lee, What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis, 2019.
10. Baoguang Shi ; Mingkun Yang ; Xinggang Wang ; Pengyuan Lyu ; Cong Yao, ASTER: An Attentional Scene Text Recognizer with Flexible Rectification, pp. 4-6, 2018.
11. Tong He, Zhi Tian, Weilin Huang, Chunhua Shen, Yu Qiao, Changming Sun, An end-to-end TextSpotter with Explicit Alignment and Attention, pp. 4-7, 2018.
12. https://github.com/tesseract-ocr/tesseract
13. 강가현, 고지현, 권용준, 권나영, 고석주, 딥러닝을 이용한 한글 OCR 정확도 향상에 대한 연구, 한국정보통신학회 2018년도 춘계학술대회, pp.693 - 695, 2018.
14. 김병희, 장병탁, 순환신경망을 이용한 한글 필기체 인식, 정보과학회 컴퓨팅의 실제 논문지 제23권 제5호, 2017.
15. https://hangeul.naver.com/2017/nanum
16. http://www.font.co.kr/yoonfont/free/main.asp
17. https://www.korean.go.kr/front/etcData/etcDataView.do?mn_id=46&etc_seq=71
18. http://www.aihub.or.kr/content/613