Towards End-to-end Text Spotting with Convolutional Recurrent Neural Networks

by jjayy

논문 링크

https://openaccess.thecvf.com/content_ICCV_2017/papers/Li_Towards_End-To-End_Text_ICCV_2017_paper.pdf

Abstract

convolutional recurrent neural networks를 바탕으로 text detection과 recognition을 한 번에 수행한다.
image cropping이나 feature re-calculation, word separation, character grouping 등의 중간 과정 없이 single forward pass로 end-to-end 학습이 가능하다.
이미지와, ground-truth, text-label만 가지고 모든 framework를 학습시킬 수 있으며, 한 번의 convolutional feature 계산으로 detection과 recognition 모두 사용하기 때문에 processing time이 적다는 장점이 있다.
여러 benchmark dataset에서 우수한 성능을 보였다.

1. Introduction

현실의 scene image는 다양한 패턴과 배경을 가지고 있기 때문에 정형화된 문서 이미지보다 더 어려운 task가 되고 있고 DNN으로 어느 정도 해결되었지만, 아직 많은 문제가 남아있다.
기존의 text detection과 recognition은 서로 다른 두 단계로 이루어져 있었는데, 실제로 text detection과 recognition은 correlate하다. 때문에 feature information이 두 단계에 있어 공유될 수 있고, 더불어 두 task가 서로를 complement하게 해줄 수 있다. (text region을 잘 찾는 것이 recognition 향상에 도움이 되고, recognition을 잘 하는 것이 detection을 refine하는데 도움)

detection과 recognition을 합쳐 spotting이라고 칭한다.

본 논문에서는 총 3가지를 제시한다.

1. convolution feature를 text detection과 recognition에 공유함으로서 여러 task를 simultaneous하게 학습할 수 있어 processing time을 줄일 수 있다. (single-end-to-end trainable network)
2. 새로운 region feature extraction. 기존의 ROI pooling layer는 region을 여러 사이즈와 aspect ratio를 고정된 사이즈로 feature map을 변환했지만, 가로로 긴 특성을 가지는 text의 특성상 추가적인 sub-optimal이 필요하다. 때문에 ROI pooling이 맞춤형으로 다양한 길이의 feature map을 생성하고 RNN encoder를 통해 동일한 사이즈로 맞춰준다.
3. learning strategy: synthetic images with simple appearance and large word lexicon to gradually more complex training data. 점점 복잡한 이미지를 학습하여 성능 향상.

2. Architecture

network architecture는 다음과 같다.
convolutional layer와 TPN, RNN encoder, MLP, attention-based RNN decoder
TPN은 text의 region을 구하기 위해, RNN encoder는 여러 fixed-length vector의 사이즈를 제안하기 위해, MLP는 detection과 bounding box reggresion을 위해, attention-based RNN decoder는 word recognition을 위해 사용했다.
이를 바탕으로 detection과 recognition이 한 번에 이루어진다.

Model

Convolutional layer은 VGG-16에서 FC layer를 제외한 부분을 사용한다. 또한, 일반적으로 text는 작기 때문에 5개의 max pooling 중 1, 2, 4번째 layer만 남겨서 down-sampling ratio를 1/32에서 1/8로 증가시켰다.
conv layer 통과 후 TPN에서 여러 scale의 sliding window를 feature map에 적용하여, local 및 contextual 정보를 얻게 된다. 이를 바탕으로 text bbox 높은 퀄리티로 얻을 수 있다.
이후, 먼저 RNN encoder(region feature encoder)는 주어진 convolutional feature를 fixed-length representation으로 변환하고 이를 MLP(Text Detection Network)에 전달한다.
MLP는 bbox offest과 textness scores를 계산한다. 이렇게 계산된 정보는 다시 RNN encoder에 전달되며, 이를 바탕으로 마지막 attention-based RNN decoder에서 text recognition을 수행합니다.

TPN(text proposal network)

RPN을 text에 맞게 변형

구조상 fully convolutional network 형태를 띄고 있다.
text의 특성상 가로가 길기 때문에 총 6개의 aspect ratio(1,2,3,5,7,10)과 4개의 scale(16, 32, 64, 80)로 총 k=24개의 anchor로 bbox를 예측합니다. 또한, W,H를 다르게 convolutional filter를 통과시켜 local과 contextual 정보 둘 다 가져갈 수 있도록 한다. 여기서 (5,3), (3, 1)과 같은 직사각형 모양의 filter를 사용하는데 이는 text의 모양과 비슷한 receptive field를 만들어 큰 aspect ratio의 정확도를 올려준다.

RNN encoder(Region Feature Encoder)

앞서 설명했던 방법대로 text의 ratio 특성에 맞춰 여러 ROI pooling을 진행하고 이렇게 구한 feature map을 LSTM에 순차적으로 넣어 key 값인 hw를 계산한다. 특히 다양한 크기의 ROI pooling으로 인해 다양한 사이즈의 feature map이 생성되는데, LSTM에 넣어주기 전 H를 fix함으로서 서로 다른 크기의 feature map이 계산되더라도 vector의 길이가 일정하도록 한다.

MLP(Text Detection Network)

앞서 Region Feature Encoder에서 LSTM을 통해 구한 hw(extracted region feature)를 바탕으로 예측한 ROI가 text인지 아닌지를 구별하고, bbox의 위치를 refine 한다. 여기서 refine 한다는 의미는 TPN에서 이미 한번 bbox를 예측했고, 이를 encoder에서 계산해서 다시 TDN에서 예측하기 때문에 refine한다고 한다.

Attention-based LSTM (Text Recognition Network)

앞서 encoder에서 얻은 hw 값들을 attection 기반의 LSTM decoder에 넣어주고, ground truth 값을 input으로 adopt하여 사용한다.

Loss function

TPN은 TDN의 bbox와 textness를 계산하는 것과 같은 것을 수행하기 때문에 loss가 다음과 같이 정의된다.
TPN loss = binary logistic loss (classification)/N + smooth_L1 loss (regression)/N(+)
N은 전체 anchor의 개수, N+는 positive anchor의 개수
여기서 postive와 negative는 mine hard negatives 방법으로 계산한다.

마지막으로 decoder에 사용된 LSTM의 cross entrophy loss 까지 함께 계산하면 total loss를 구할 수 있다.

total loss = binary logistic loss (classification)/N + smooth_L1 loss (regression)/N+ + cross entrophy loss/N+ (recognition)

learning curriculm

1. 48k images containing words in the Generic lexicon of size 90k (lock TRN initially for 30k)
2. after 30k train TRN with lr of 10-3
3. for next 50k, used Synth800K dataaset with average 10 synthetic words placed on each real scene background
4. 2044 real-world images from ICDAR2015 dataset