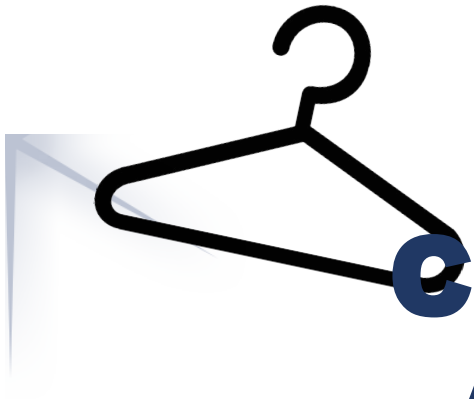


Creating a Closet using Artificial intelligence

인공지능을 활용한 나만의 옷장 만들기 서비스

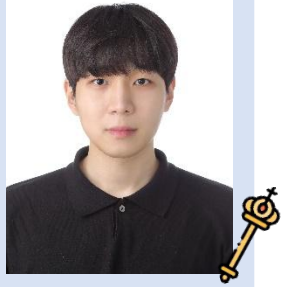
CLOZET

Team

팀장 한성수

개발/기획/배포

담당 : 스프링 개발 및 아마존 배포



팀원 김윤섭

개발/기획/배포

담당 : 스프링 개발 및 아마존 배포

팀원 김승현

개발/기획/배포

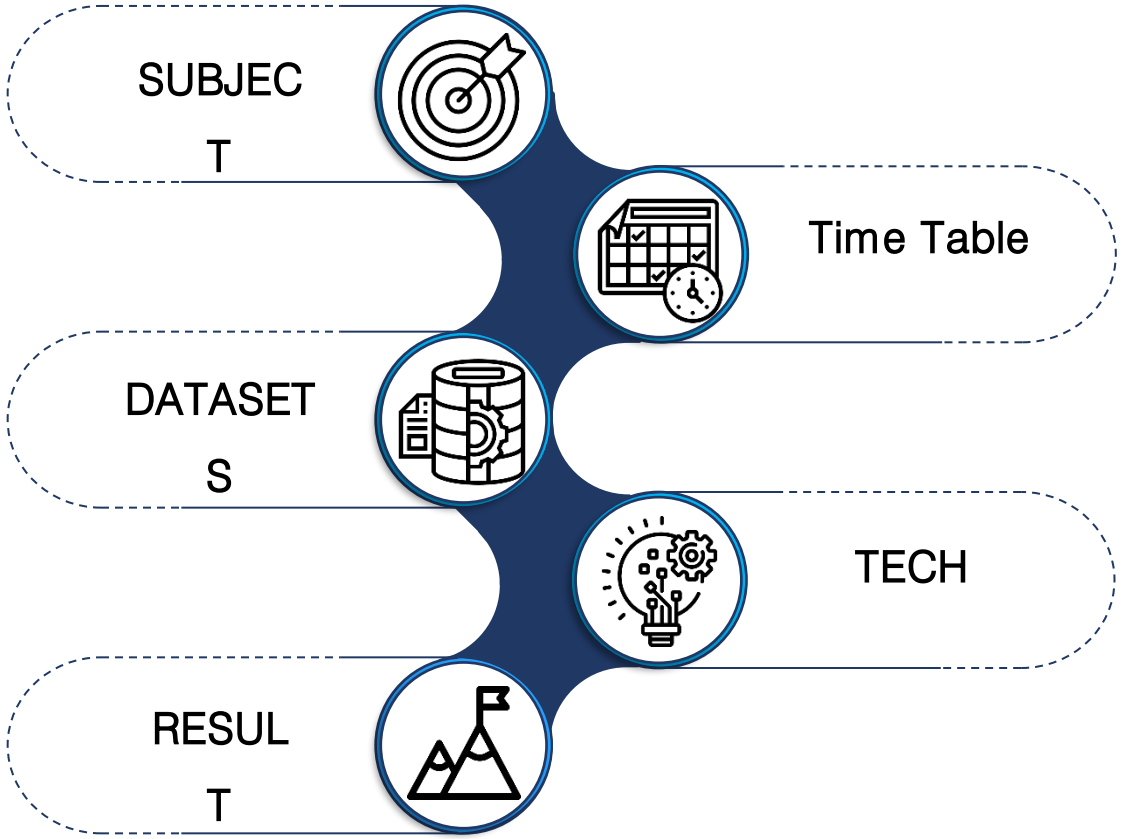
담당 : 의류 및 색상 인식인공지능 개발

팀원 유재혁

개발/기획/배포

담당 : 웹 페이지 개발 및 아마존 배포

Category

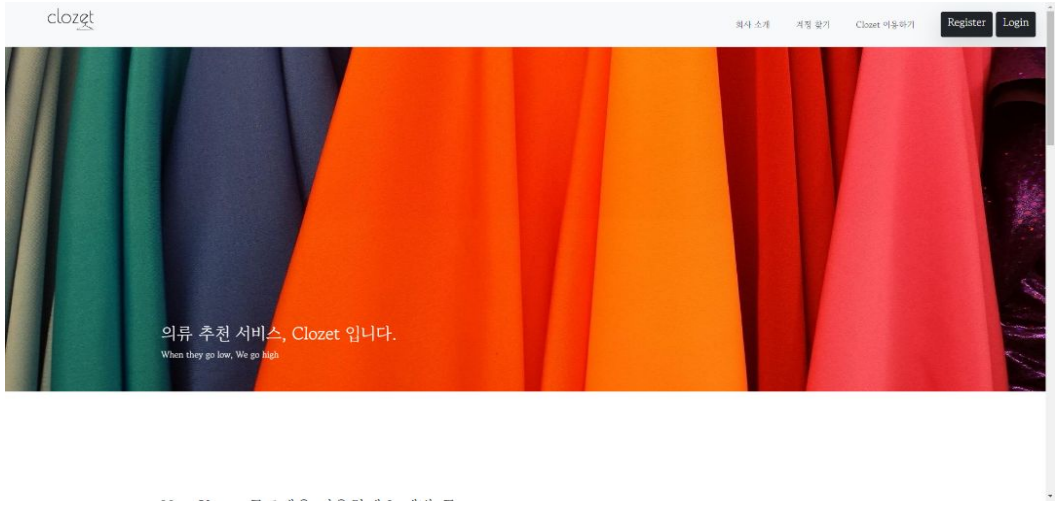


Subject

Basic Concept

AI × Spring × React

AI **Spring** **React**



<5>

Time Table

상세설계/정보수집
스터디/논문분석
Data Setting /전처리, EDA
실구축
테스팅

MAY | JUNE | JULY | AUGUST

21~ 01 06 13 20 27 03 10 17 27 01 08

OCR

Translation

최종점검/ 플랫폼 구축

How & What to Study

팀 모델 설정

Idea sharing & Decision

E

Github 공유 Repository

보강 작업

팀원 간 소통. 부족한 부분 보강

D

A

논문 분석

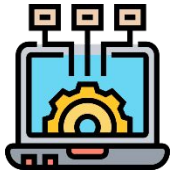
필요 논문 탐색 및 분석

C

B

기술 정리

논문 분석 → 기술 구현 → 기술정리



기술 구현

구현에 필요한 기술 스터디

Get Datasets

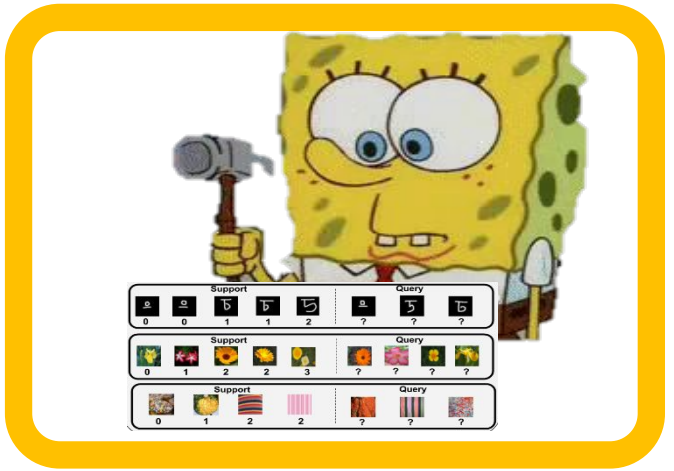
ICDAR Datasets

ICDAR 2013/2015/2017

Datasets

SYNTHTEXT DATA

90k datasets



Self-made Datasets

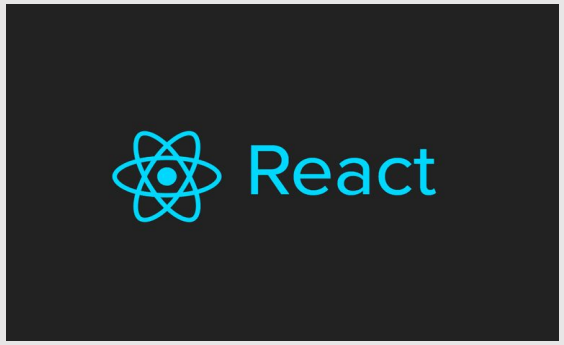
직접 촬영, 데이터 증폭 등

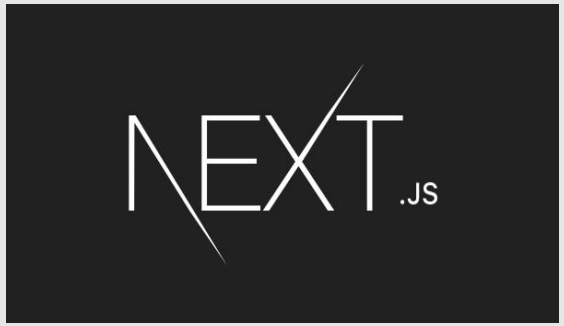
Train
Image data

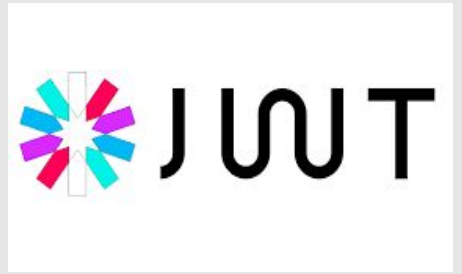
Eval
Image data

Data
sets

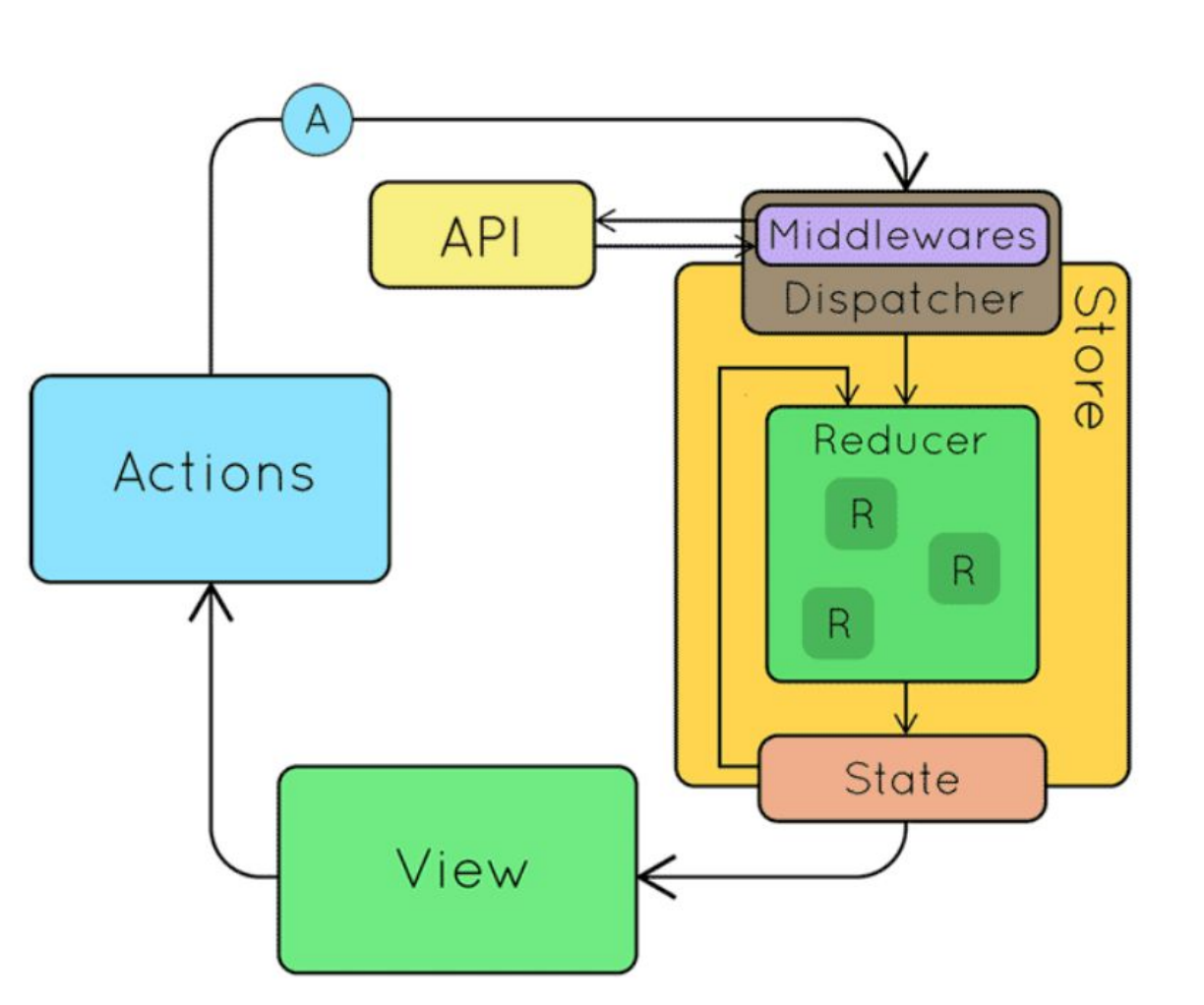
Frontend Technology Stack







Flux Pattern

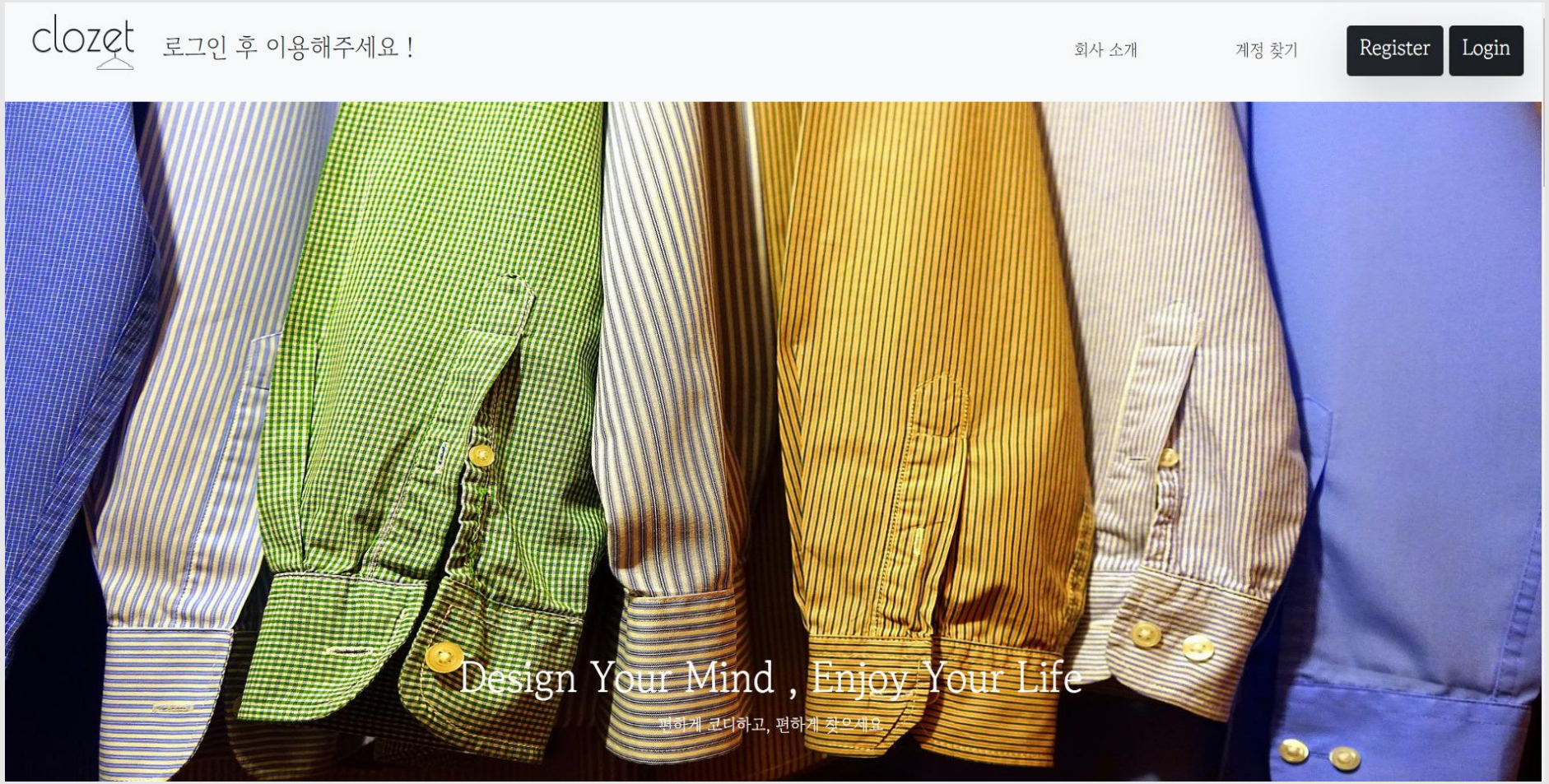


React - Redux는 Flux 패턴으로 움직입니다.
MVC 패턴처럼 보일 수도 있지만,

클라이언트의 Action을 Reducer Slice를 통해 이동해, Action이 생성됩니다.

Action은 Store에 Action내의 Payload와 State 를 저장하며, Saga를 통해 API를 호출해 Spring과의 HTTP Method를 통한 REST를 가능하게 합니다.

BootStrap 활용한 반응형 웹 UI/UX 화면 생성



```tsx
useEffect(() => {
  const token : string | null = localStorage.getItem("loginSuccessUser")
  if ( token !== null) {
    setInfo({token : token})
  } else {
    throw('error')}
  },[date])

const dispatch = useAppDispatch()

const onInfoChange = (e: React.FormEvent<HTMLInputElement>) => {
  e.preventDefault()
  const { name , value } = e.currentTarget
  setInfo({ ...info , [name]: value})
}

const onInfoSubmit = (e: React.FormEvent<HTMLFormElement>) => {
  e.preventDefault()
  dispatch(updateRequest(info))
}

const onDeleteClick = (e: React.MouseEvent<HTMLButtonElement>) => {
  e.preventDefault()
  console.log(`토큰 전송 : ${JSON.stringify(info.token)}`)
  dispatch(removeRequest(info.token))
}
```

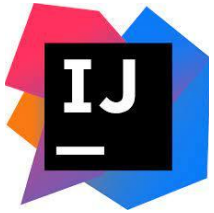
브라우저 스토리지를 이용한 로그인 상태 유지
JSON Web Token을 Spring으로부터 발급받아, 인가를 처리한다.

Techs

Backend Technology Stack

PuTTY

ubuntu

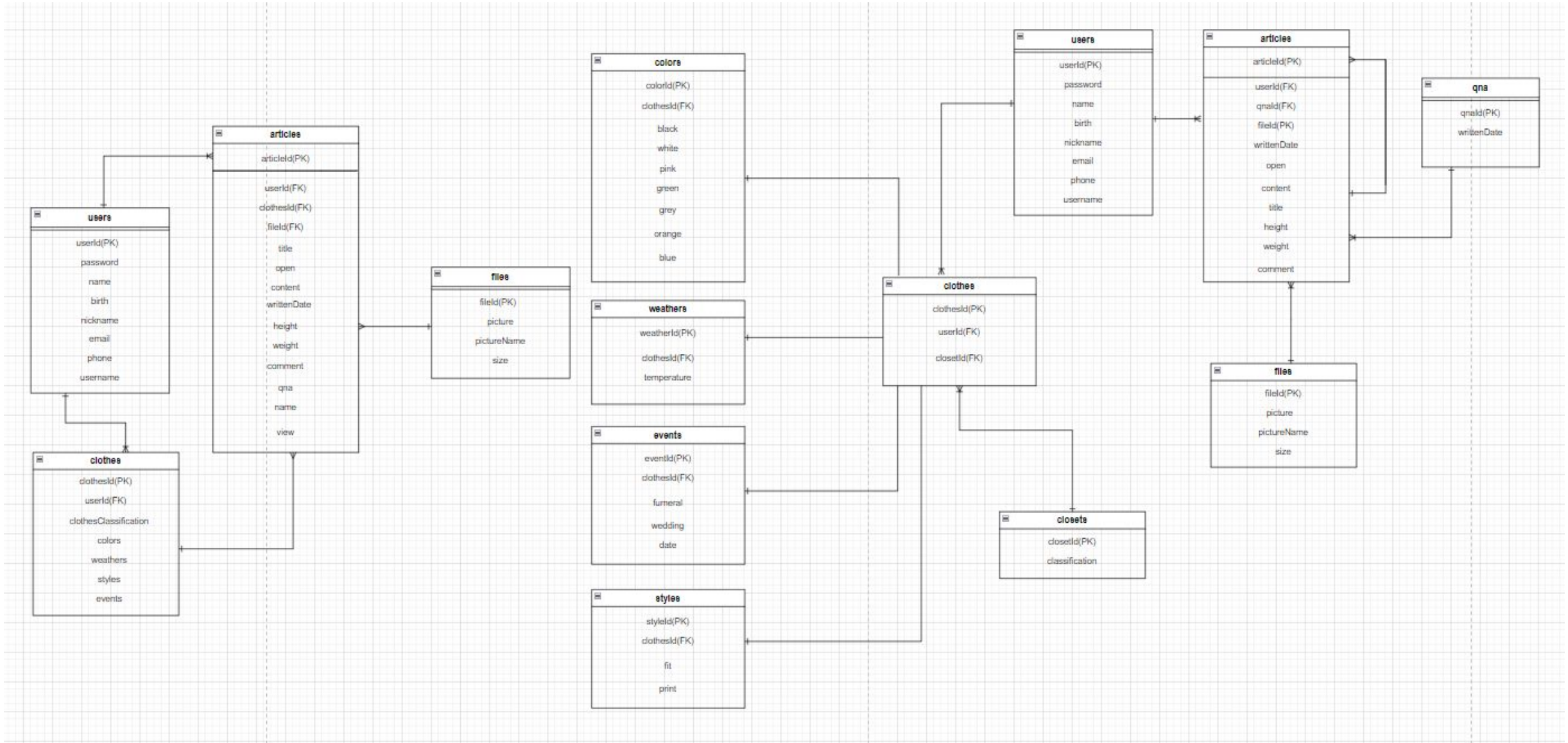


POSTMAN

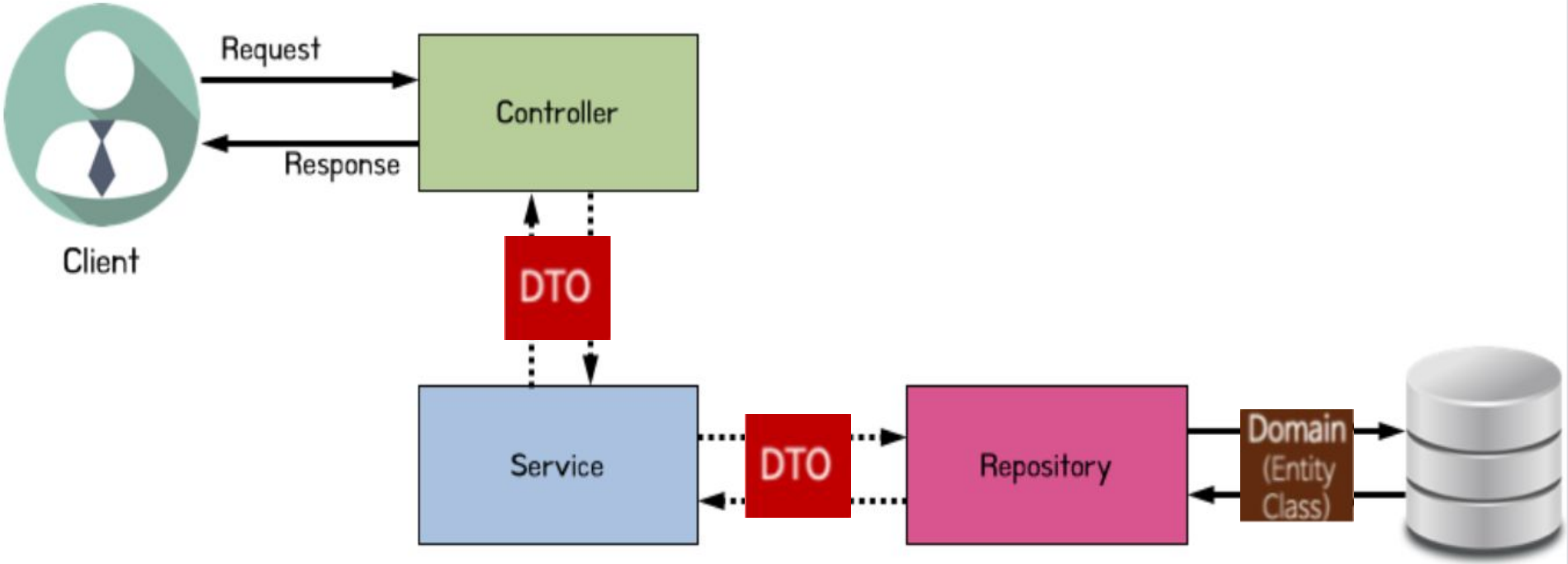
Apache Tomcat

ERD

- **ERD**의 정규화와 반정규화



- **MVC Pattern**


Request
Response
Client
Controller
DTO
Service
DTO
Repository
Domain
(Entity
Class)

Techs

Frame works

TensorFlow

OpenCV

Steps

DETECTION

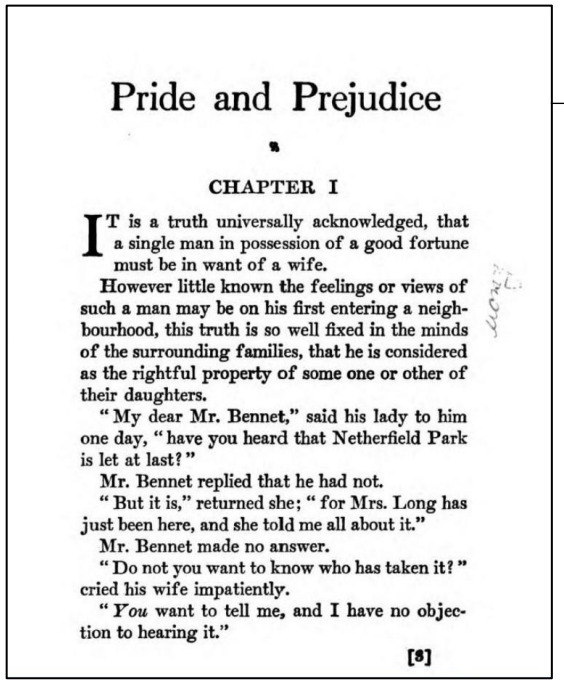
Recognition

TRANSFORMER

• Classic OCR과의 차이점

Classi c OCR

Scene Text Detection And Recognition



Pride and Prejudice

CHAPTER I

IT is a truth universally acknowledged, that a single man in possession of a good fortune must be in want of a wife.

However little known the feelings or views of such a man may be on his first entering a neighbourhood, this truth is so well fixed in the minds of the surrounding families, that he is considered as the rightful property of some one or other of their daughters.

"My dear Mr. Bennet," said his lady to him one day, "have you heard that Netherfield Park is let at last?"

Mr. Bennet replied that he had not.

"But it is," returned she; "for Mrs. Long has just been here, and she told me all about it."

Mr. Bennet made no answer.

"Do not you want to know who has taken it?" cried his wife impatiently.

"*You* want to tell me, and I have no objection to hearing it."

[3]

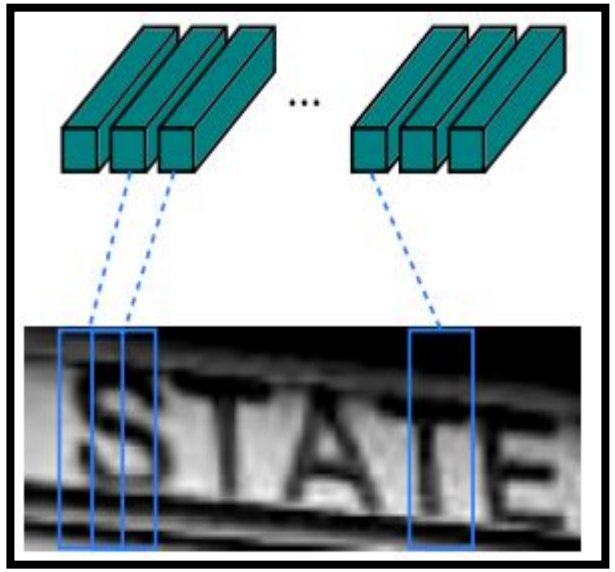
OCR		Scene text
단순함	배경	복잡함
규칙적	글씨체	다양함
수평	글자 배열	각도 및 구도가 다양함
단조로움	색	다채로움

☰ Overview

Detection

EAST: An Efficient and Accurate Scene Text Detector

Recognition



CRNN: An End-to-End Trainable Neural Network for Image

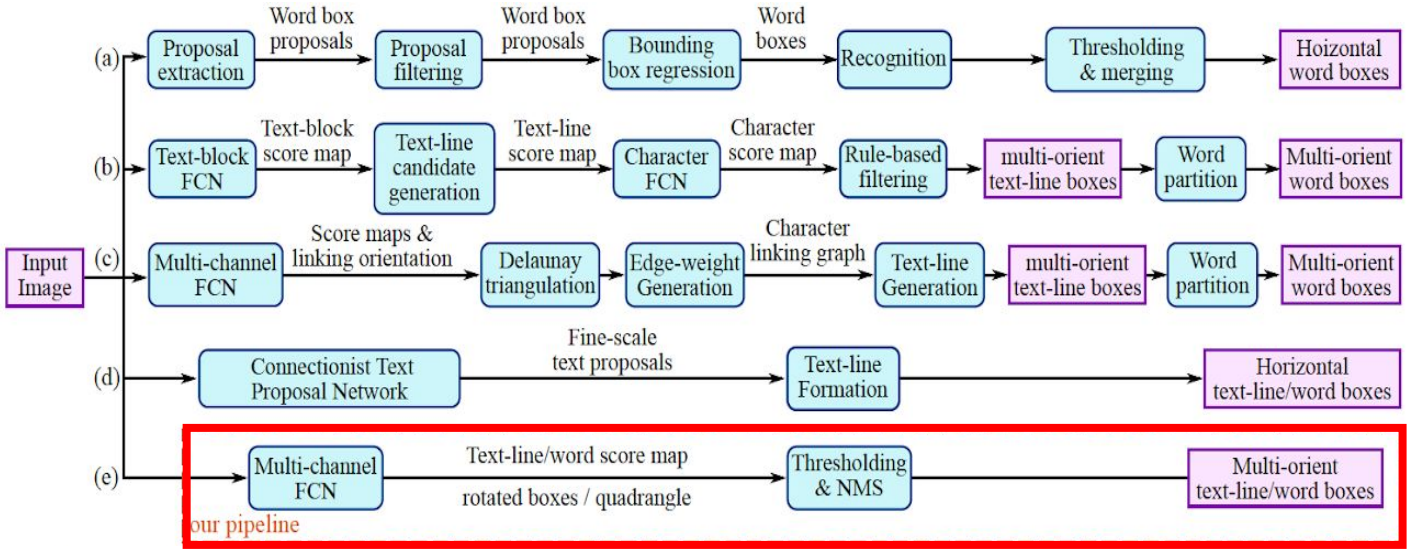
Text Spotting

FOTS : Fast Oriented Text Spotting

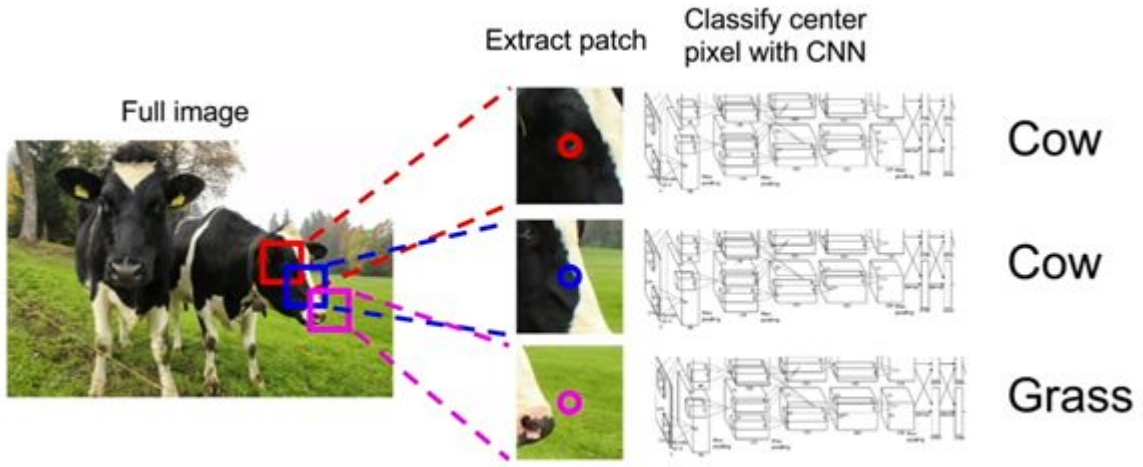
Detection

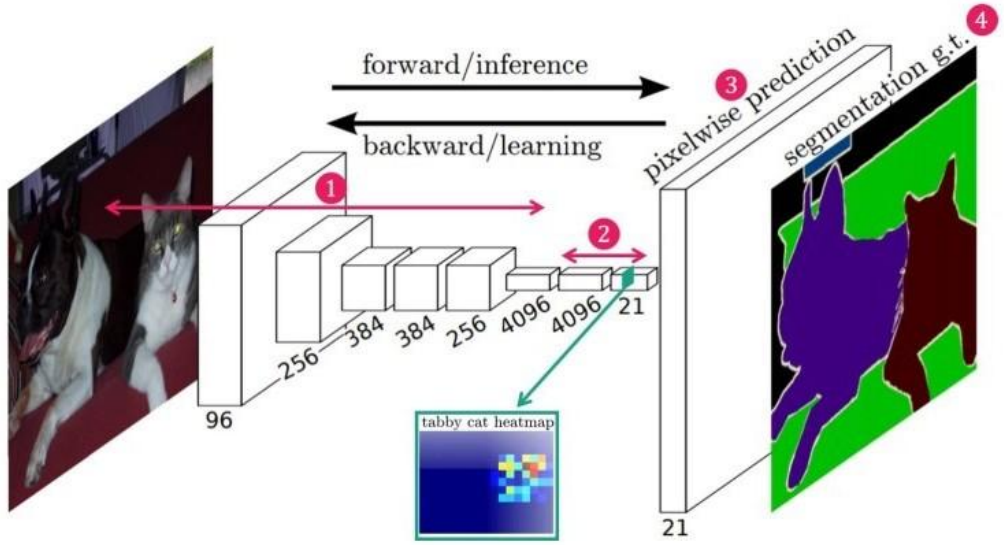
EAST: An Efficient and Accurate Scene Text Detector(Zhou. Et al.,2017)

- 기존 Text detection 모델들보다 연산 과정을 줄여**시간을 대폭 단축함**
- FCN(Fully Convolution Network)알고리즘 활용 > **정확도 증가**

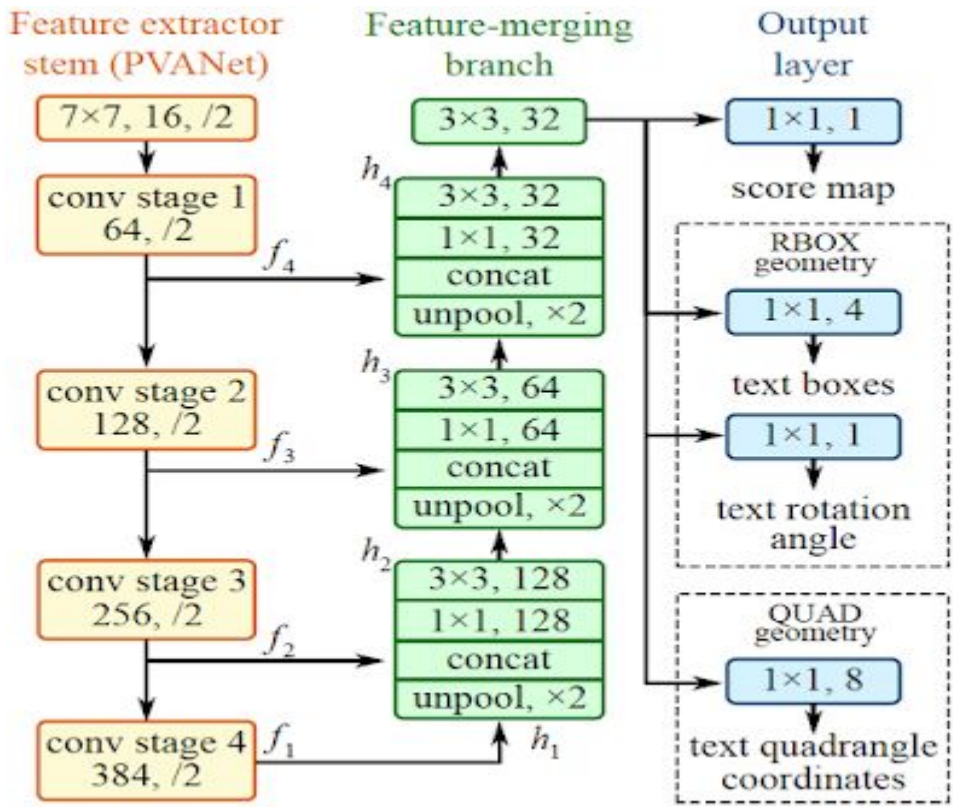


Fully Convolutional Network(FCN)





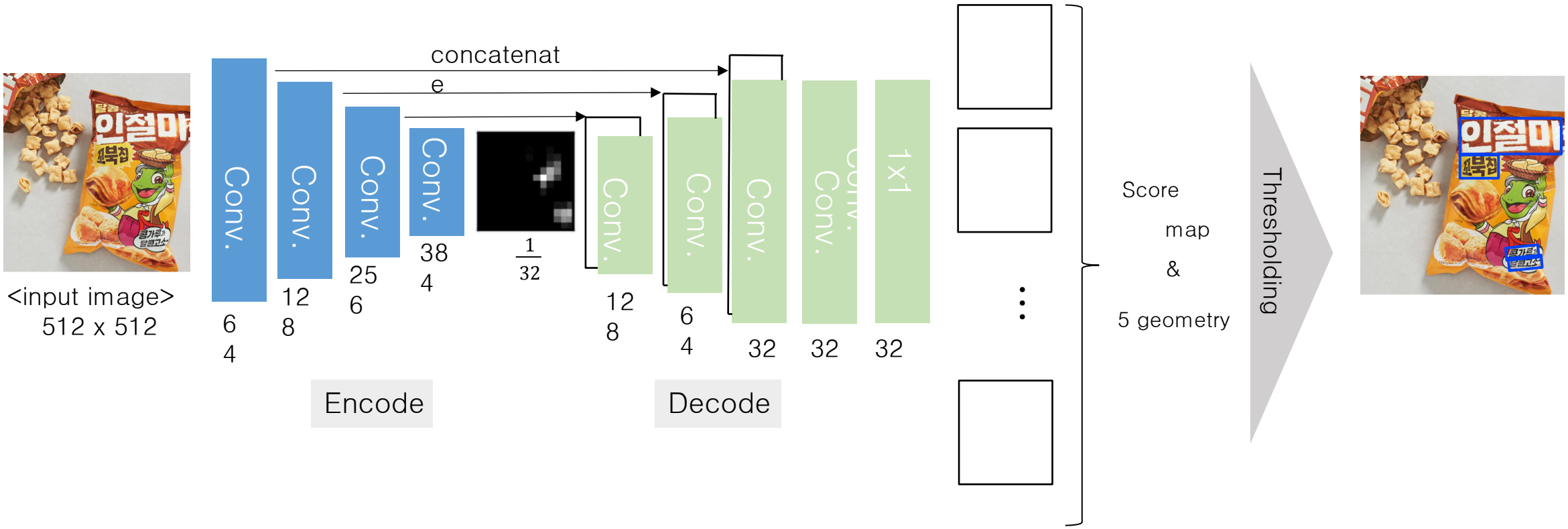
EAST : Architecture, score & geo



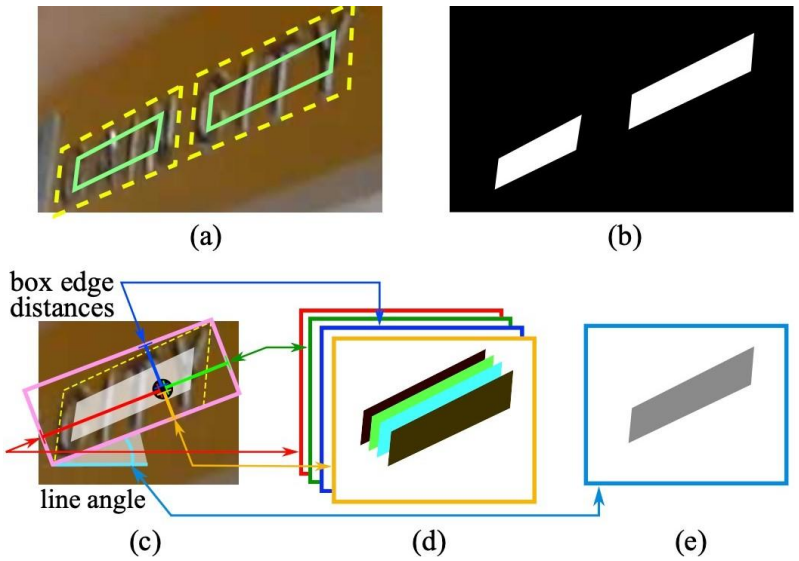
Score map(text의 확률을 0~1값으로 표현)

Geo map(x1, y1, w, h, 각도)

EAST : Structure



EAST : Loss



Thresholding

$$L = L_s + \lambda_g L_g$$

예측 label

실제 label

- [Score Loss]

  Positive & Nagative (밝음 & 어두움)으로 구성되는 score를 추정하기 위한 loss

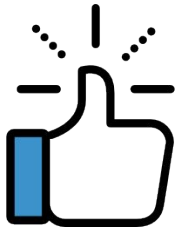
- [Geometry Loss]

  임의 사각형의 4개 꼭지점을 추정하기 위해 RBOX 및 QUAD 정보에 설정하는 loss

전체 손실 함수

$$L = L_s + \lambda_g L_g$$

EAST : Detection format



목표

▶ 자유도 8의 QUAD 사각형

RECT 직사각형
Rectangle
자유도 4 : (x,y,w,h)

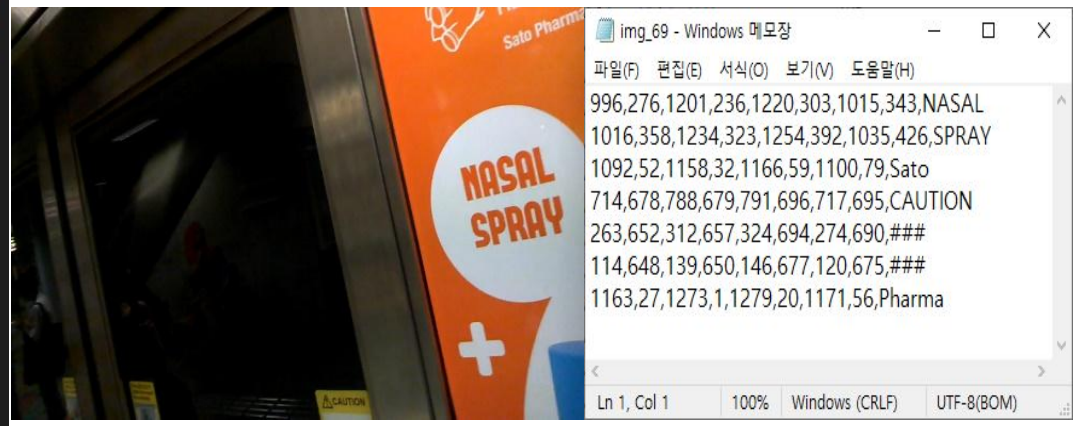
RBOX 직사각형+각도
Rotated Box
자유도 5 : (x,y,w,h,θ)

QUAD 사각형
quadrilateral
자유도 8 :
(x1,y1,x2,y2,X3,y3,x4,y4)

POLY 다각형
Polygon
N개의 점으로 표현 시
자유도 2N

EAST : data_preprocess

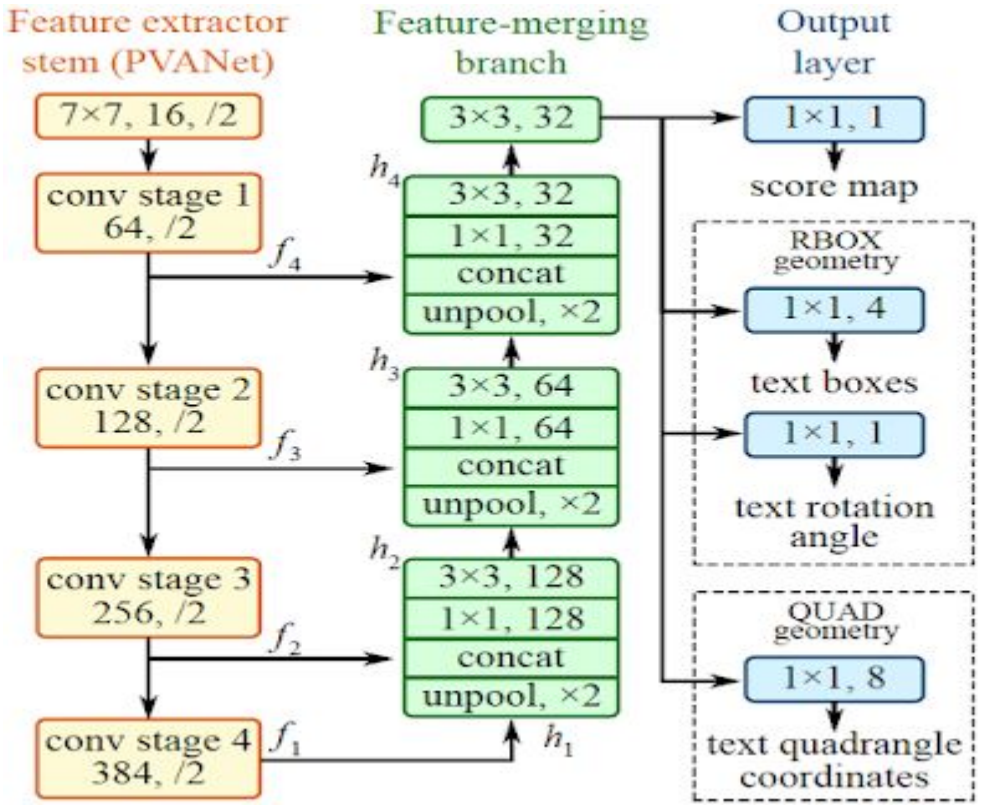
```python
class custom_dataset(data.Dataset):
    def __init__(self, img_path, gt_path, scale=0.25, length=512):
        super(custom_dataset, self).__init__()
        self.img_files = [os.path.join(img_path, img_file) for img_file in sorted(os.listdir(img_path))]
        self.gt_files  = [os.path.join(gt_path, gt_file) for gt_file in sorted(os.listdir(gt_path))]
        self.scale = scale
        self.length = length

    def __len__(self):
        return len(self.img_files)

    def __getitem__(self, index):
        with open(self.gt_files[index], 'r', encoding='UTF-8') as f:
            lines = f.readlines()
        vertices, labels = extract_vertices(lines)

        img = Image.open(self.img_files[index])
        img, vertices = adjust_height(img, vertices)
        img, vertices = rotate_img(img, vertices)
        img, vertices = crop_img(img, vertices, labels, self.length)
        transform = transforms.Compose([transforms.ColorJitter(0.5, 0.5, 0.5, 0.25), \
                                        transforms.ToTensor(), \
                                        transforms.Normalize(mean=(0.5,0.5,0.5),std=(0.5,0.5,0.5))])

        score_map, geo_map, ignored_map = get_score_geo(img, vertices, labels, self.scale, self.length)
        return transform(img), score_map, geo_map, ignored_map
```



Img, score_map,
geo_map, ignored_map

핵심 모델 구성

```
path = os.path.dirname(os.path.realpath(__file__))

class EAST(nn.Module):
    def __init__(self, pretrained=True):
        super(EAST, self).__init__()
        self.extractor = extractor(pretrained)
        self.merge     = merge()
        self.output    = output()

    def forward(self, x):
        return self.output(self.merge(self.extractor(x)))
```



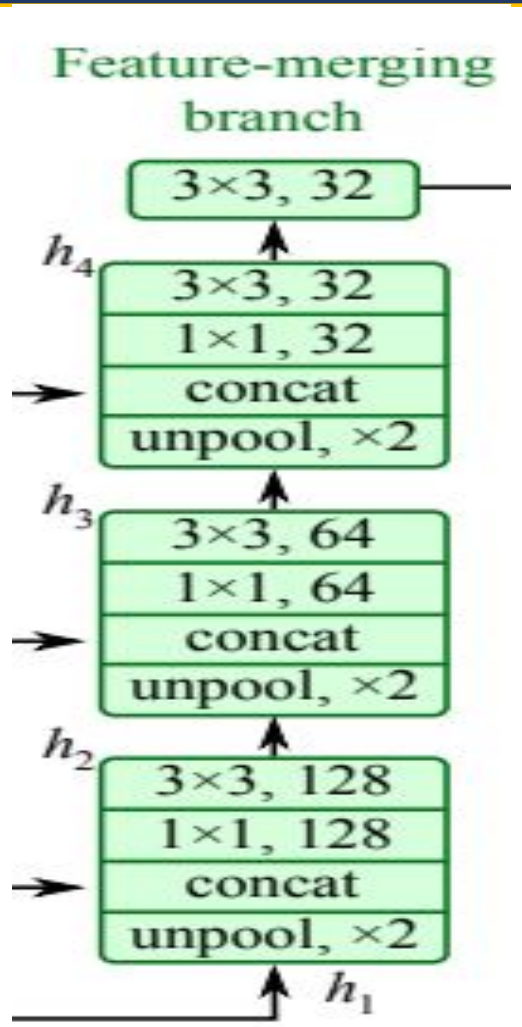
☰ EAST : extractor

```
class extractor(nn.Module):
    def __init__(self, pretrained):
        super(extractor, self).__init__()
        vgg16_bn = VGG(make_layers(cfg, batch_norm=True))
        if pretrained:
            vgg16_bn.load_state_dict(torch.load(f'{path}/pths/vgg16.pth'))
        self.features = vgg16_bn.features

    def forward(self, x):
        out = []
        for m in self.features:
            x = m(x)
            if isinstance(m, nn.MaxPool2d):
                out.append(x)
        return out[1:]
```

Layer를 생성하고 훈련을 실행하여 Feature를 추출해주는 class

EAST : merge

```
def forward(self, x):
    y = F.interpolate(x[3], scale_factor=2, mode='bilinear', align_corners=True)
    y = torch.cat((y, x[2]), 1)
    y = self.relu1(self.bn1(self.conv1(y)))
    y = self.relu2(self.bn2(self.conv2(y)))

    y = F.interpolate(y, scale_factor=2, mode='bilinear', align_corners=True)
    y = torch.cat((y, x[1]), 1)
    y = self.relu3(self.bn3(self.conv3(y)))
    y = self.relu4(self.bn4(self.conv4(y)))

    y = F.interpolate(y, scale_factor=2, mode='bilinear', align_corners=True)
    y = torch.cat((y, x[0]), 1)
    y = self.relu5(self.bn5(self.conv5(y)))
    y = self.relu6(self.bn6(self.conv6(y)))
    y = self.relu7(self.bn7(self.conv7(y)))

    return y
```



이미지를 원래의 사이즈로 다시 복원시킨다.

EAST : output

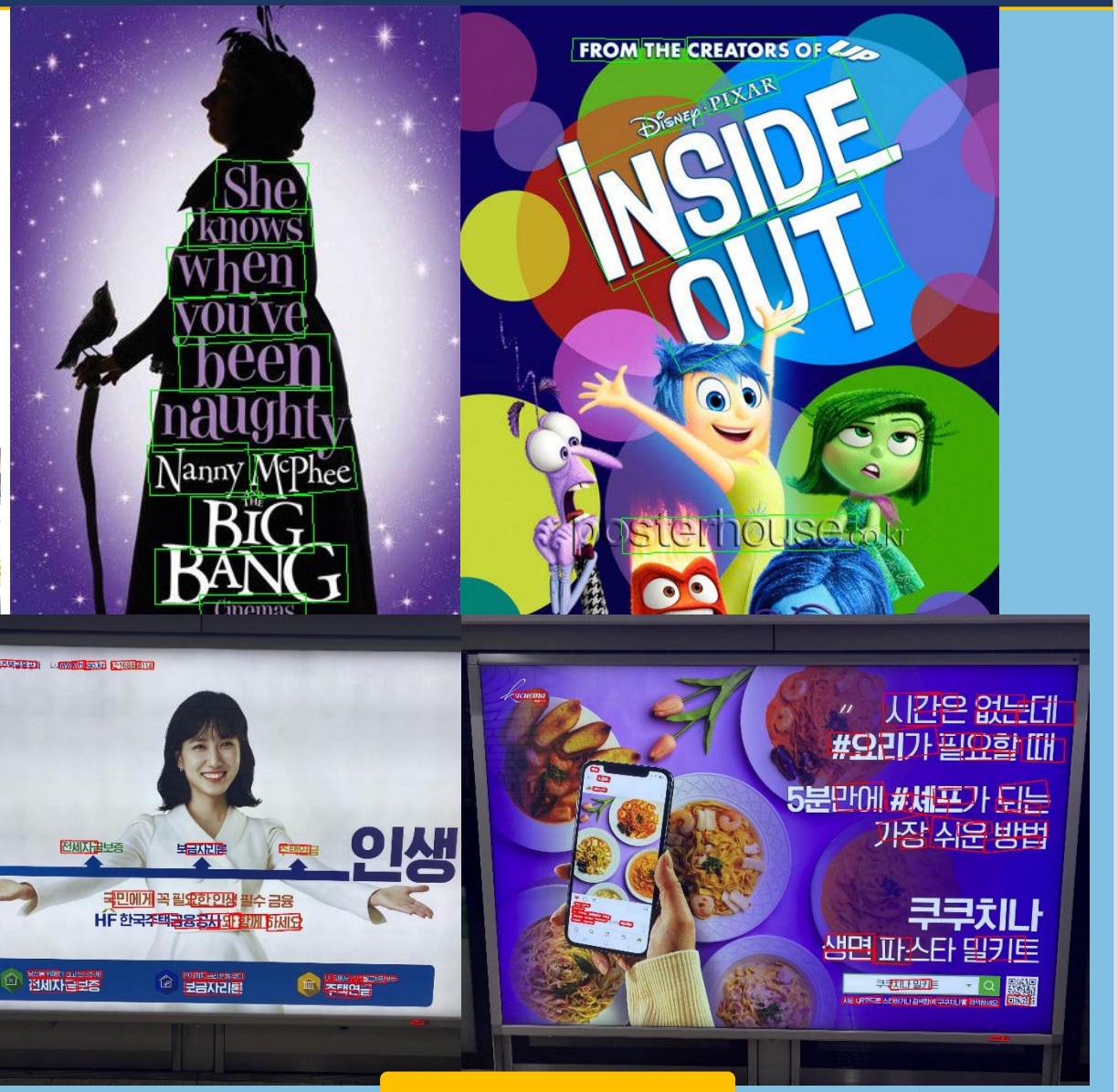
```
class output(nn.Module):
    def __init__(self, scope=512):
        super(output, self).__init__()
        self.conv1 = nn.Conv2d(32, 1, 1)
        self.sigmoid1 = nn.Sigmoid()
        self.conv2 = nn.Conv2d(32, 4, 1)
        self.sigmoid2 = nn.Sigmoid()
        self.conv3 = nn.Conv2d(32, 1, 1)
        self.sigmoid3 = nn.Sigmoid()
        self.scope = 512
        for m in self.modules():
            if isinstance(m, nn.Conv2d):
                nn.init.kaiming_normal_(m.weight, mode='fan_out', nonlinearity='relu')
                if m.bias is not None:
                    nn.init.constant_(m.bias, 0)

    def forward(self, x):
        score = self.sigmoid1(self.conv1(x))
        loc   = self.sigmoid2(self.conv2(x)) * self.scope
        angle = (self.sigmoid3(self.conv3(x)) - 0.5) * math.pi
        geo   = torch.cat((loc, angle), 1)
        return score, geo
```

Score = 0 ~ 1

Geo = (x1 ,y1 ,w,h, θ)

EAST : Eval Output

ICDAR DATA SET



Optional DATA SET

EAST : Eval Output comparison


```python
if __name__ == '__main__':
    train_img_path = os.path.abspath('D:\OCR\EAST-master_3\ICDAR 2015 Challenge 4\ch4_training_images')
    train_gt_path  = os.path.abspath('D:\OCR\EAST-master_3\ICDAR 2015 Challenge 4\ch4_training_localization_transcription_gt')
    pths_path      = './pths'
    batch_size     = 8
    lr             = 1e-4
    num_workers    = 2
    epoch_iter     = 600
    save_interval  = 100
    train(train_img_path, train_gt_path, pths_path, batch_size, lr, num_workers, epoch_iter, save_interval)
```

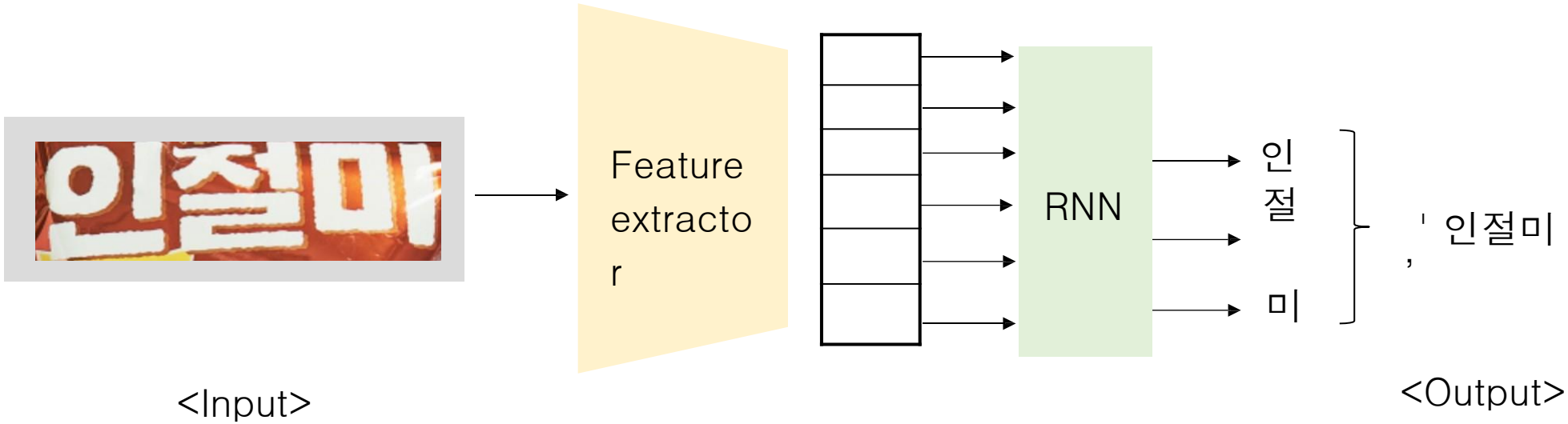
```python
if __name__ == '__main__':
    train_img_path = os.path.abspath('D:\TeamProjectCode\EAST-master_3\ICDAR 2015 Challenge 4\ch4_training_images')
    train_gt_path  = os.path.abspath('D:\TeamProjectCode\EAST-master_3\ICDAR 2015 Challenge 4\ch4_training_localization_transcription_gt')
    pths_path      = './pths'
    batch_size     = 64
    lr             = 1e-3
    num_workers    = 8
    epoch_iter     = 700
    save_interval  = 5
    train(train_img_path, train_gt_path, pths_path, batch_size, lr, num_workers, epoch_iter, save_interval)
```

Comparison of tuning results

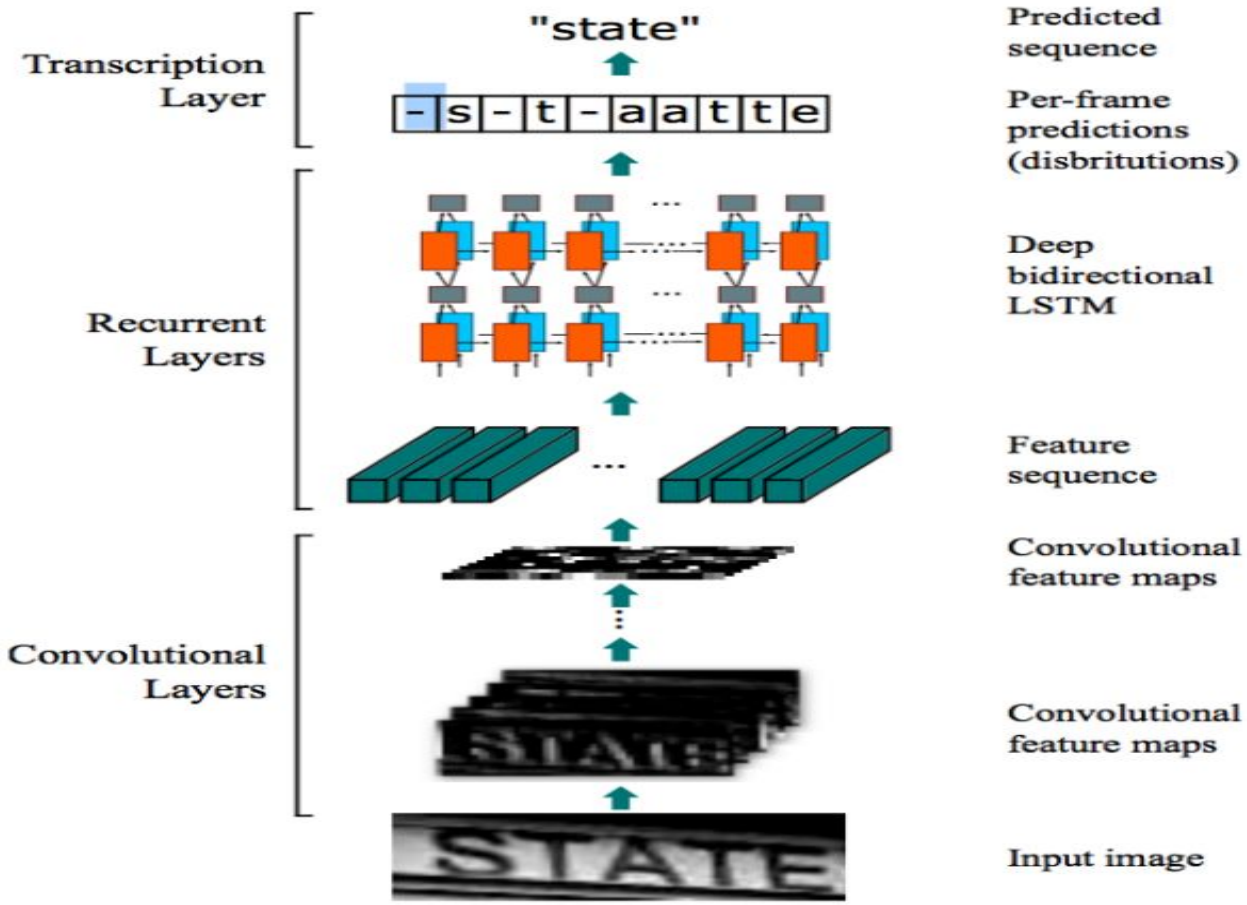
Recognition

CRNN: An End-to-End Trainable Neural Network for Image Sequence Recognition and Its Application to Scene Text Recognition(Shi. Et al., 2015)

- 각 단어 영역이 어떤 문자인지 찾는 Classification model
- 단어 영역에 해당하는 이미지로부터 특징 추출 후 sequential하게 만들어 각 글자의 조합(단어)을 찾아가는 방식
- CNN과 RNN을 결합해 Scene Text Recognition 문제를 해결한 초기 모델



☰ CRNN : Architecture



• (Step 4) CTC 알고리즘을 통해 중복 문자와 공백 등 을 제거한 후 최종 단어 예측 값 출력

• (Step 3) 길이가 긴 시퀀스 입력을 gradient vanishing 문제 없이 처리할 수 있는 bi-LSTM 모델을 이용해, 각 벡터에 대한 글자 예측 값 출력

• (Step 2) 추출된 feature map을 열 벡터 단위로 나눠 시퀀스 형태로 변환

• (Step 1) Convolution 연산을 통해 특징 추출

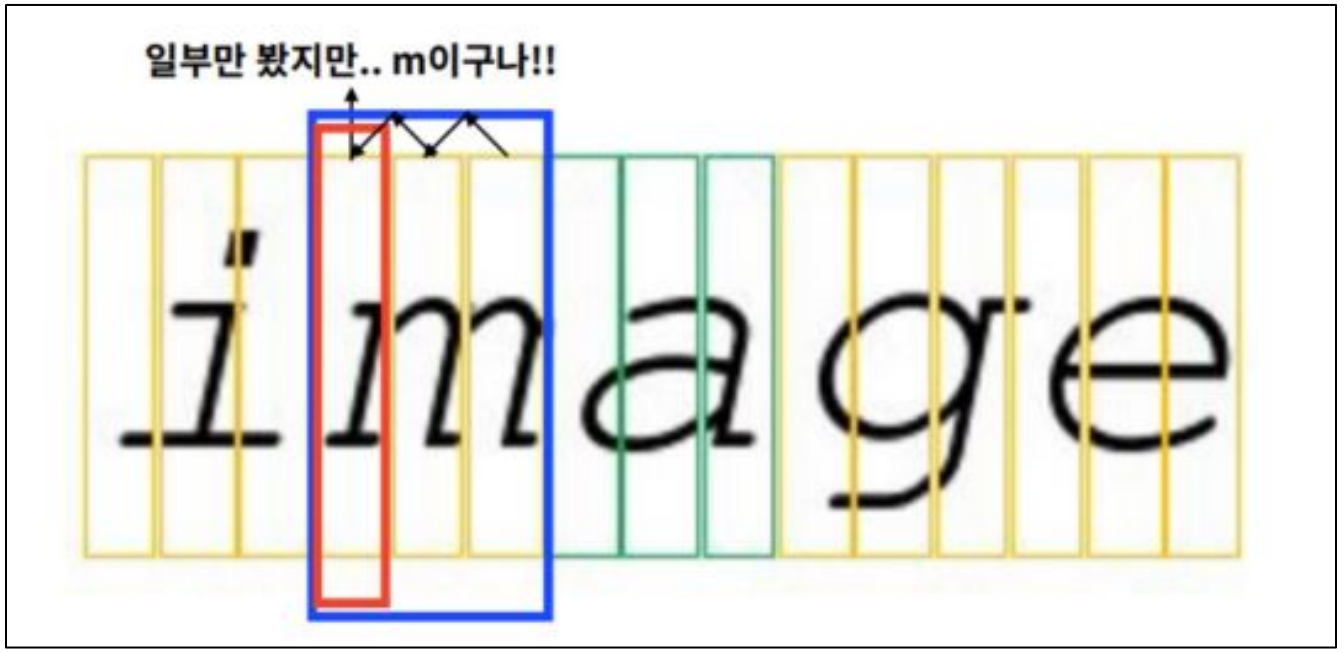
☰ CRNN : RNN

Why RNN?

CRNN = CNN + RNN

- 이미지마다 글자의 크기, 배치가 다르기 때문에 글자 단위로 정확하게 나누는 것은 어려움
- feature map의 부분적 정보만을 이용해 글자를 예측하려면 앞뒤의 다른 정보를 종합적으로 고려해야 함

(예시) m의 일부는 i일 수도 l일 수도 있지만 시퀀스 내 다른 정보를 고려해 m으로 판단 가능

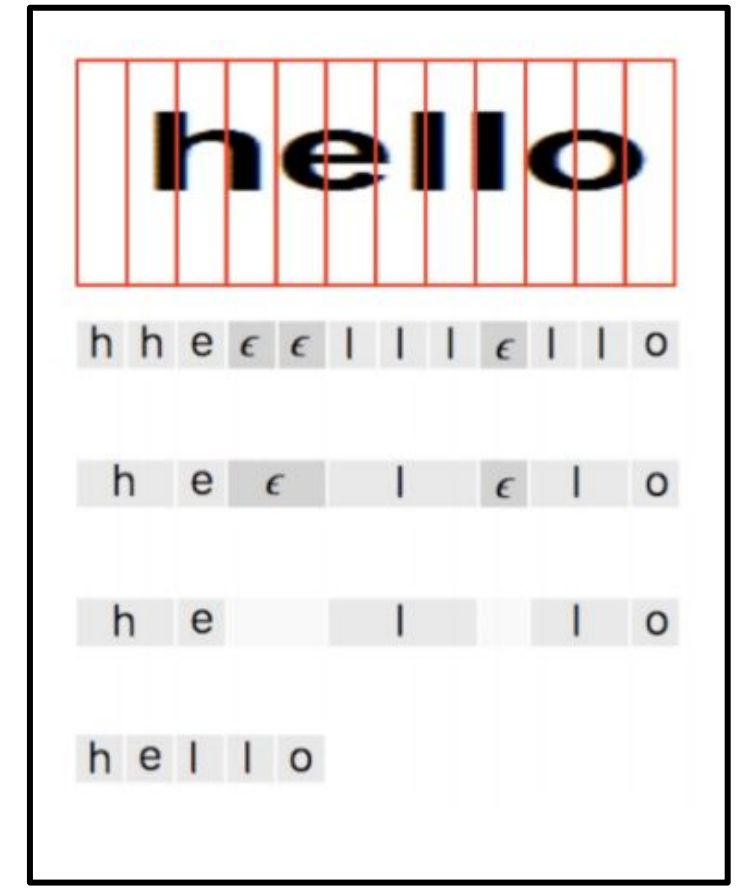


☰ CRNN : CTC Loss

Connectionist Temporal Classification(CTC)

- 음성인식 모델에서 사용되는 알고리즘으로, 시퀀스 분류 모델 맨 마지막 에 손실 및 그래디언트 계산 레이어 역할을 함

- '–' 를 활용하여, 단어 사전에 저장되어 있는 path를 찾을 수 있도록
  한 것이 특징



CRNN : Source code

```
# Recognition Model
def recognition():
    input = Input(shape = (64,128,3), dtype='float32',name ='input')
    x = conv_bn_relu(input,64,(3,3),1,'same')
    x = MaxPool2D(pool_size = (2, 1),strides=(2,1),padding='same')(x)
    x = conv_bn_relu(x,64,(3,3),1,'same')
    x = MaxPool2D(pool_size = (2, 1),strides=(2,1),padding='same')(x)
    x = conv_bn_relu(x,32,(3,3),1,'same')
    x = conv_bn_relu(x,32,(3,3),1,'same')
    x = MaxPool2D(pool_size = (2, 1),strides=(2,1),padding='same')(x)

    x = conv_bn_relu(x,32,(3,3),1,'same')
    x = conv_bn_relu(x,32,(3,3),1,'same')
    x = MaxPool2D(pool_size = (2, 1),strides=(2,1),padding='same')(x)


    x = conv_bn_relu(x,64,(3,3),1,'same')
    x = Reshape(target_shape= ((64,-1)))(x)
    x= Dense(64, activation='relu', kernel_initializer='he_normal')(x)
```

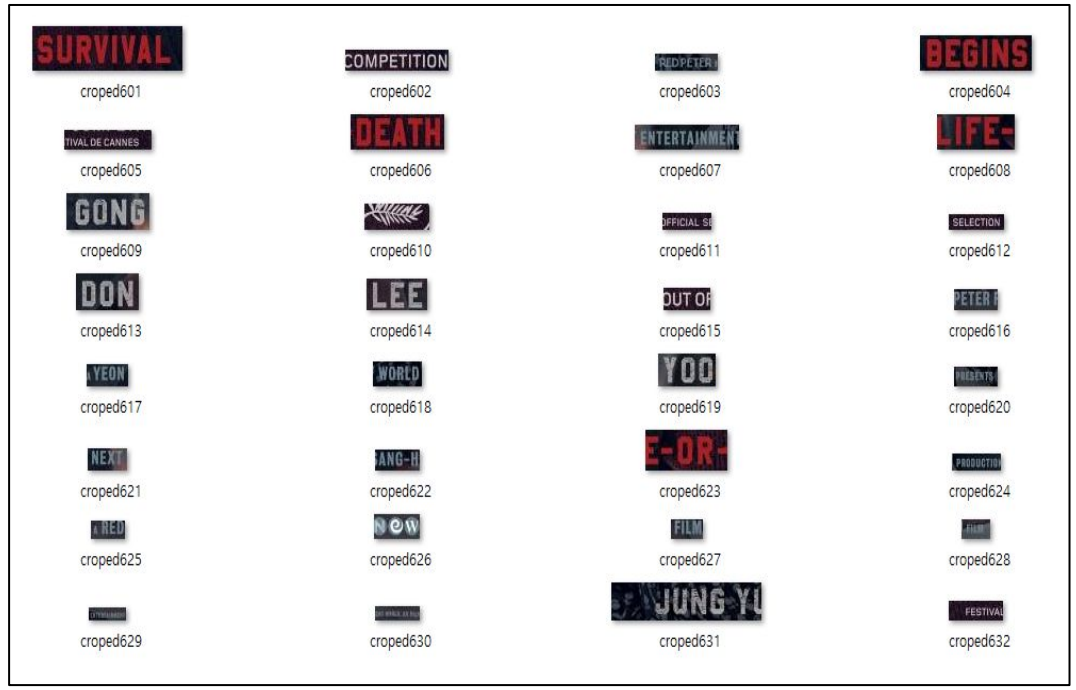
```
x= Dropout(rate=0.1)(x)
x = Bidirectional(LSTM(128,return_sequences=True,go_backwards=True))(x)

x= Dropout(rate=0.1)(x)
x = Bidirectional(LSTM(128,return_sequences=True,go_backwards=True))(x)

x= Dropout(rate=0.1)(x)
out = Dense(units=n_classes+2,activation='softmax',
kernel_initializer=tf.keras.initializers.glorot_normal(seed=0))(x)

recognizer=Model(input, out)

return recognizer
```

CRNN : crop image

Self-made modeling

1. Text Detection
2. Image crop
3. Text Recognition

CRNN : crop image



CRNN : output

```
croped601.png recognition : ss---u--r--v--i-v-aa-l---- => survival
croped602.png recognition : c--o--m--p-e--t-it-ioo-n-- => competition
croped603.png recognition : r----e--d--p-e--t-e--r--t- => redpetert
croped604.png recognition : b-----ee--g---ii-n---ss--- => begins
croped605.png recognition : t--m-al---e-c-a--h-e-si-s- => tmalecahesis
croped606.png recognition : d-------e---a----t---h---- => death
croped607.png recognition : e---n-t-er-t-aiin--m-e-ntt => entertainment
croped608.png recognition : l-----i---f-----e-----s--- => lifes
croped609.png recognition : g--------o-----n----g----- => gong
croped610.png recognition : s-----a---m-----n---e----- => samne
croped611.png recognition : o--f-f--i-c-iiaa-l----s--- => officials
croped612.png recognition : s----e--l-e-c--t-i-o-n---- => selection
croped613.png recognition : d----------o-------n------ => don
croped614.png recognition : l----------e------ee------ => lee
croped615.png recognition : o-----u-----t-----o-----p- => outop
croped616.png recognition : p----e---t---e----r----y-- => petery
croped617.png recognition : p-----y---e---o-----n----- => pyeon
croped618.png recognition : w--------o---r---l---d---- => world
croped619.png recognition : y-----------o-------o----- => yoo
croped620.png recognition : p---u--e--s--e-n--t--s---- => puesents
croped621.png recognition : nn-------e----x----t------ => next
croped622.png recognition : a-------n---g----t---h---- => angth
croped623.png recognition : f----------o-----r----s--- => fors
croped624.png recognition : p----r--o-d--u--c-t-i-onn- => production
croped625.png recognition : w--------r-----e----d----- => wred
croped626.png recognition : n---------e--------w------ => new
croped627.png recognition : ff-------i---l----m------- => film
croped628.png recognition : f--------i--------l---s--- => fils
croped629.png recognition : c----------t---l--a-t--s-- => ctlats
croped630.png recognition : m-----o---l--t--i-n---g--- => molting
croped631.png recognition : s----o--u--u--n--g--l-y-l- => souunglyl
croped632.png recognition : p----------e-s--t-i-n----- => pestin
```

Text Spotting

Detection

\+

Recognition

=

Text Spotting

‘Original’, ‘JP.CHENET’, ‘FRANCE’, ‘BLANCDEBLANCS’, ‘SWEET’

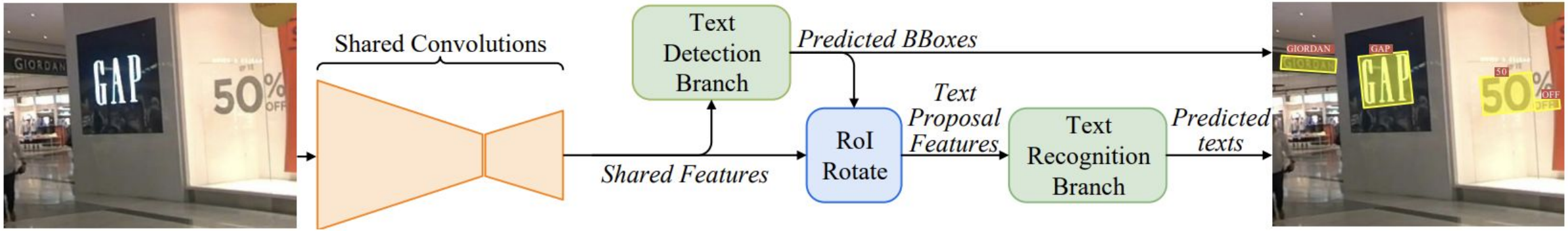
Text Spotting

FOTS : Fast Oriented Text Spotting with Unified Network(Liu. et al., 2018)

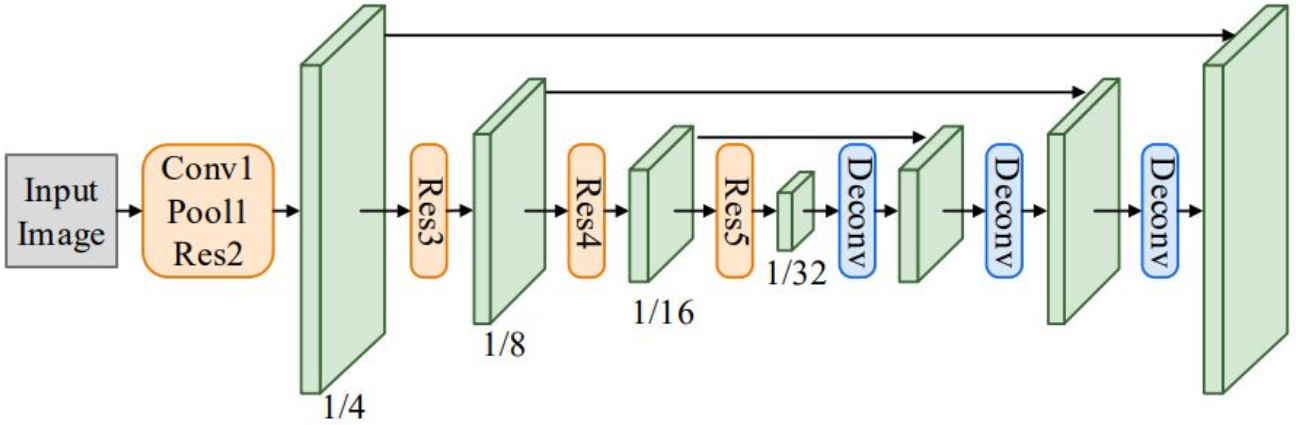
- EAST와 CRNN을 하나로 합친 모델, 단순히 모델을 붙인 것이 아닌, 한 번의 특징 추출로 Detection과 Recognition을 수행함으로, 연산 시간을 크게 줄임

Detection + Recogniton = 83.5 ms <-> **Text Spotting(FOTS) = 44.2 ms 약 2배의 속도 차이**

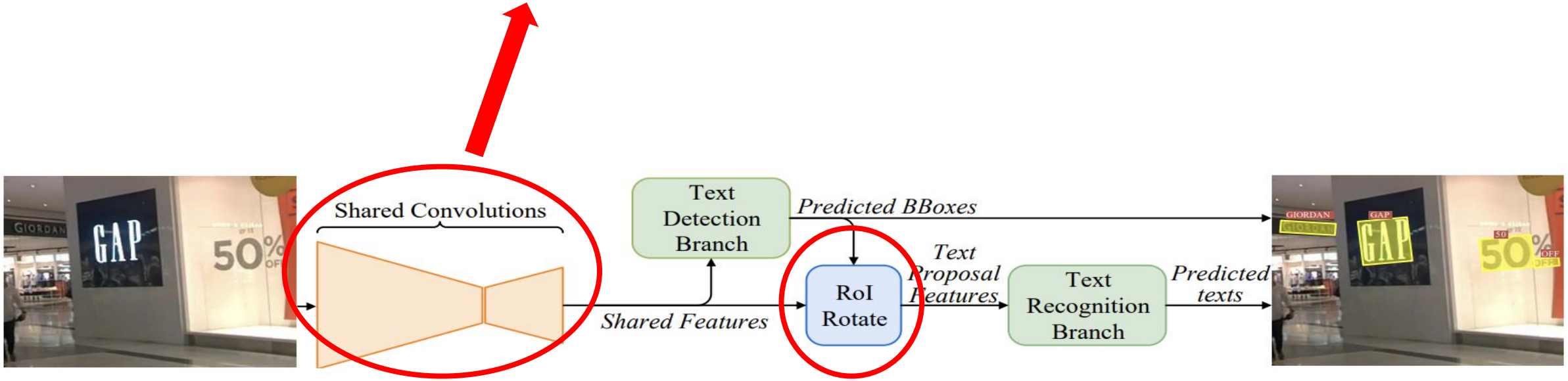
- RoI Rotate를 활용



FOTS : (1) Shared Convolution



- ResNet – 50을 사용
- Feature Map를 ¼ 사이즈로 추출

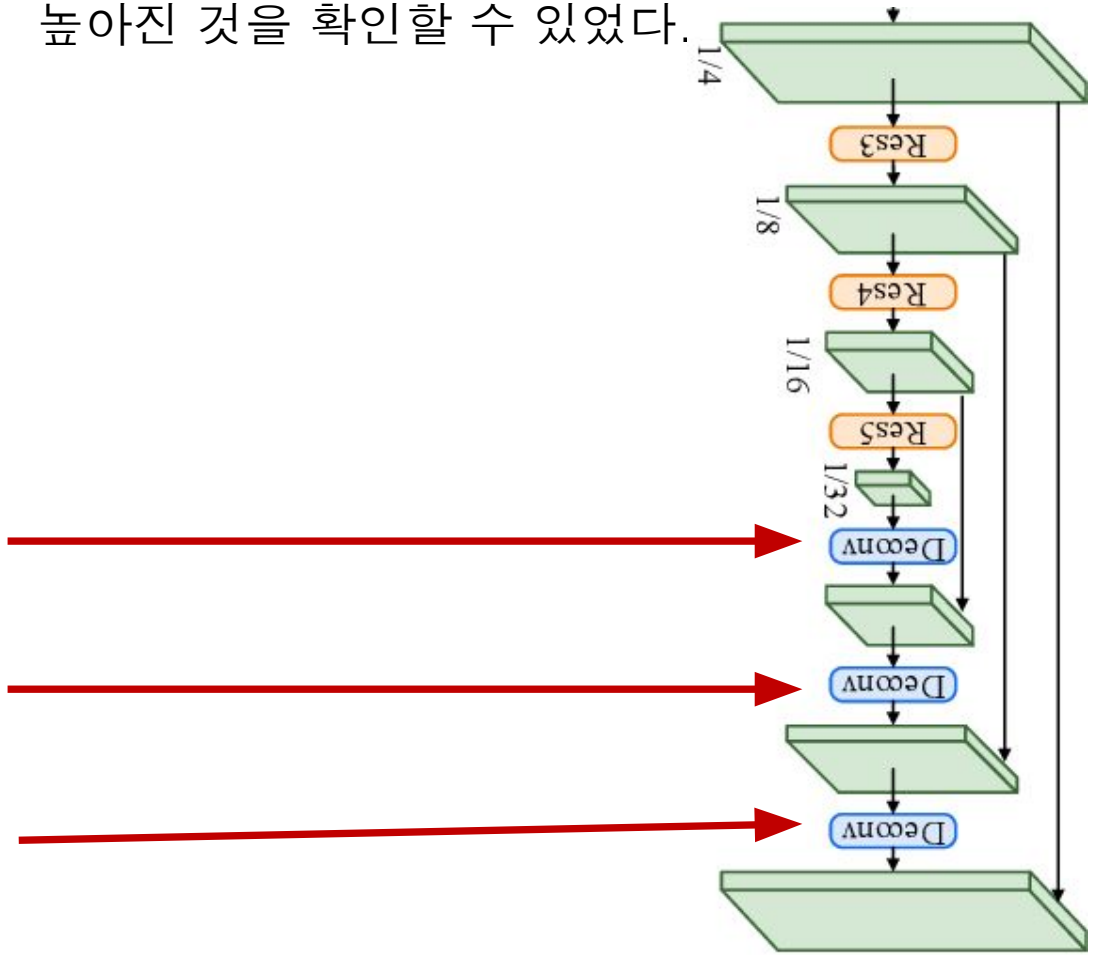


FOTS : (1) Shared Convolution

```
# Shared network

resnet=tf.keras.applications.ResNet50(input_shape=(512,512,3),
                                      include_top=False,
                                      weights='imagenet')
tf.keras.backend.clear_session()

for layers in resnet.layers:
    layers.trainable=False



x = resnet.get_layer('conv5_block3_out').output

x = Deconv('Deconv1')(x)
x = tf.keras.layers.add([x,resnet.get_layer('conv4_block6_out').output])

x = Deconv('Deconv2')(x)
x = tf.keras.layers.add([x,resnet.get_layer('conv3_block4_out').output])

x = Deconv('Deconv3')(x)
x = tf.keras.layers.add([x,resnet.get_layer('conv2_block3_out').output])
```

- Detection과 Recognition이 공유하는 피쳐맵을 추출

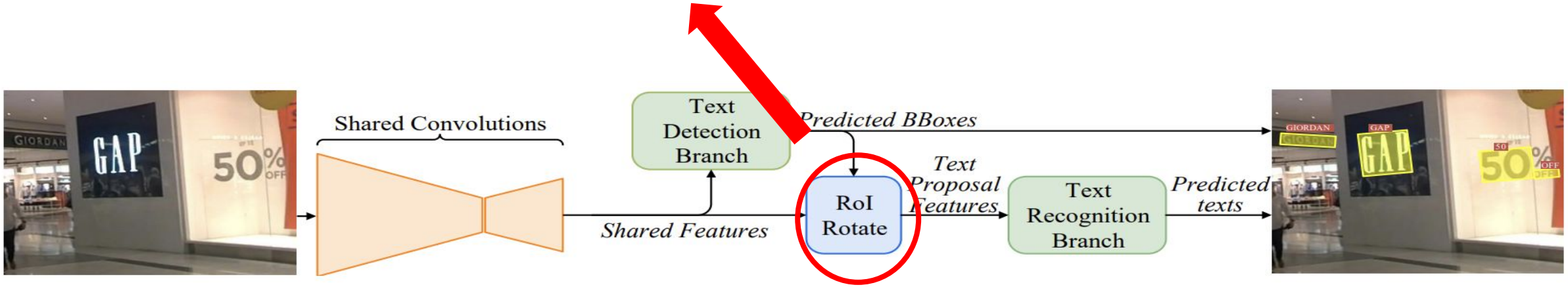
- Shared Convolution을 사용했더니 정확도가 높아진 것을 확인할 수 있었다.



FOTS : (2) RoI rotate

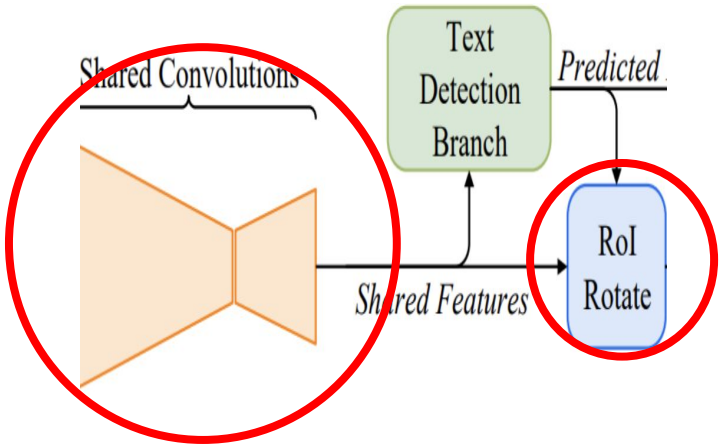
- 높이를 8로 고정
- affine transformer 사용하여 각도 조정
- 넓이는 padding을 사용하여, 일정한 크기로 조정

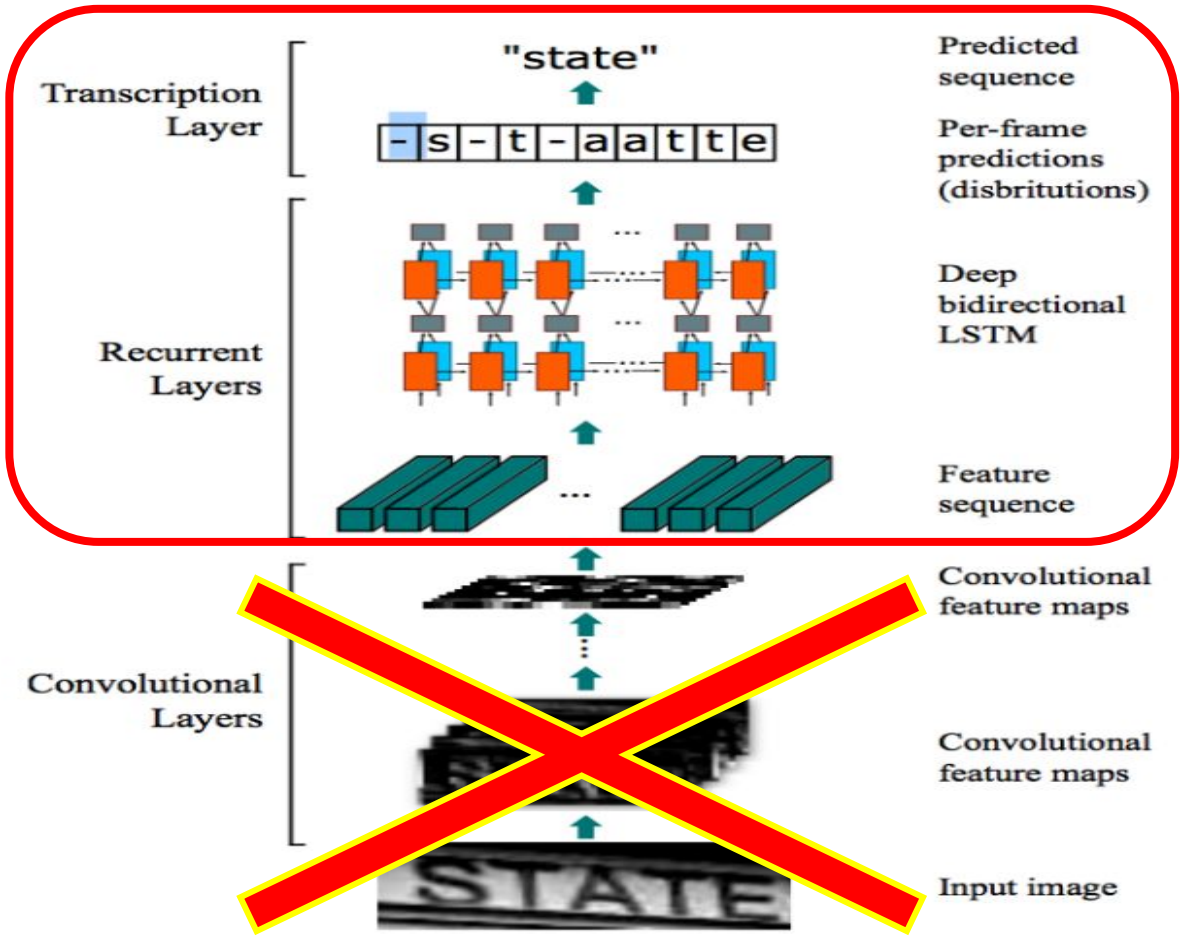
Oriented Text region box > Canonical box



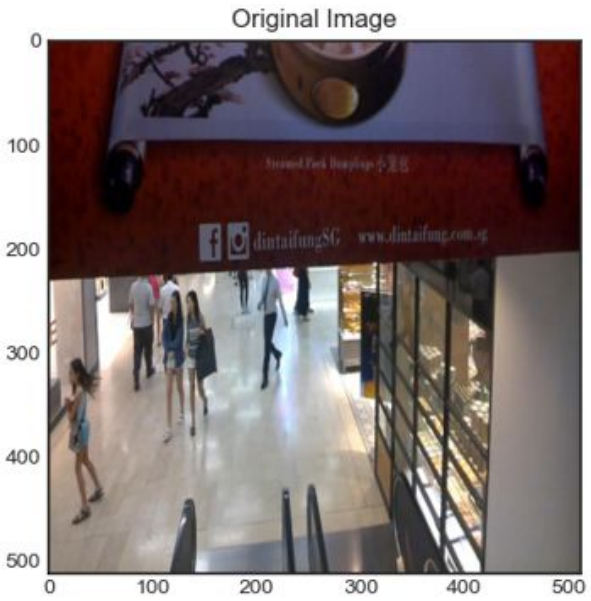
FOTS : (2) RoI rotate

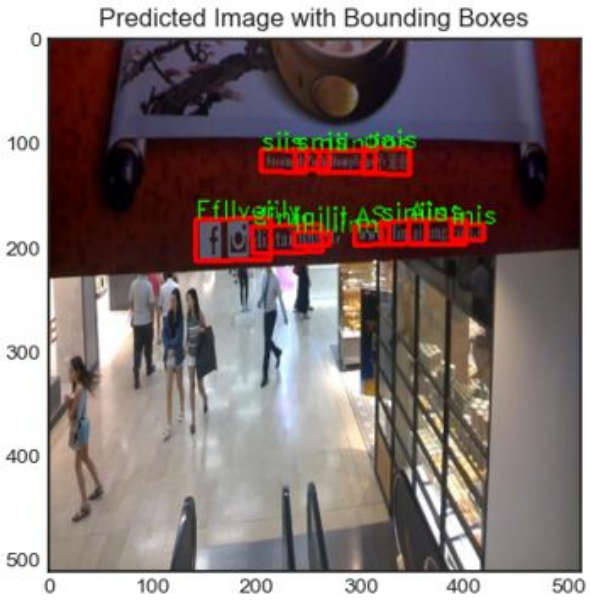
Shared Convolution + RoI Rotate

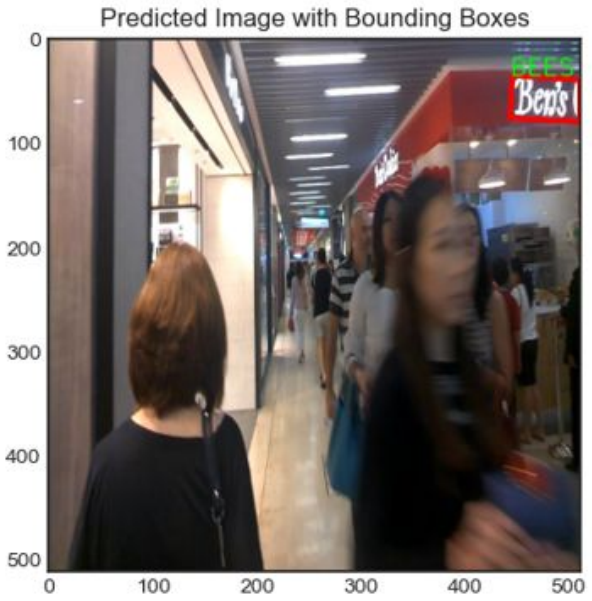


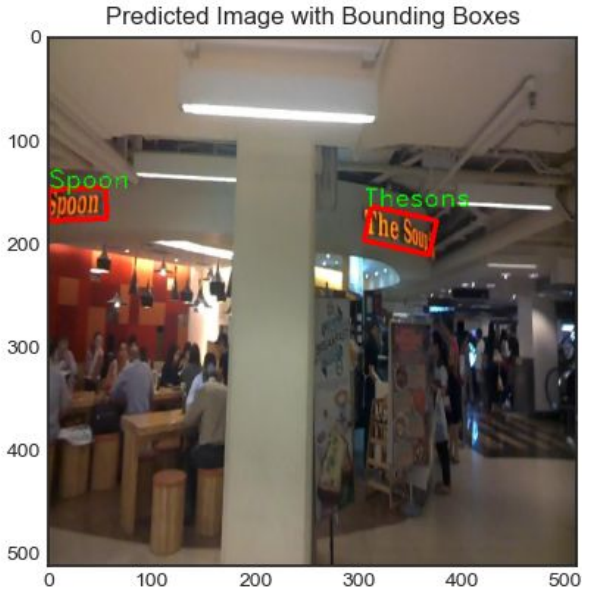


FOTS : Eval Output comparison 1

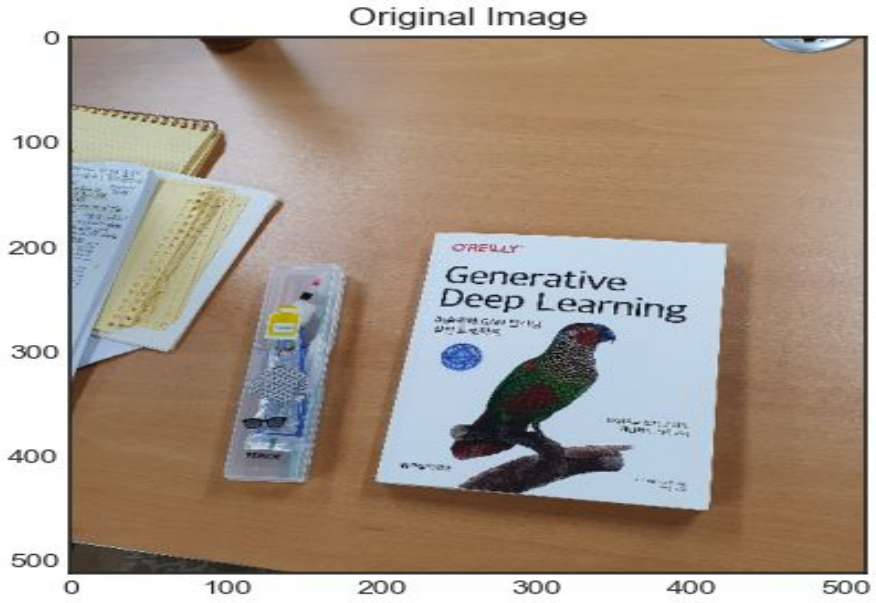


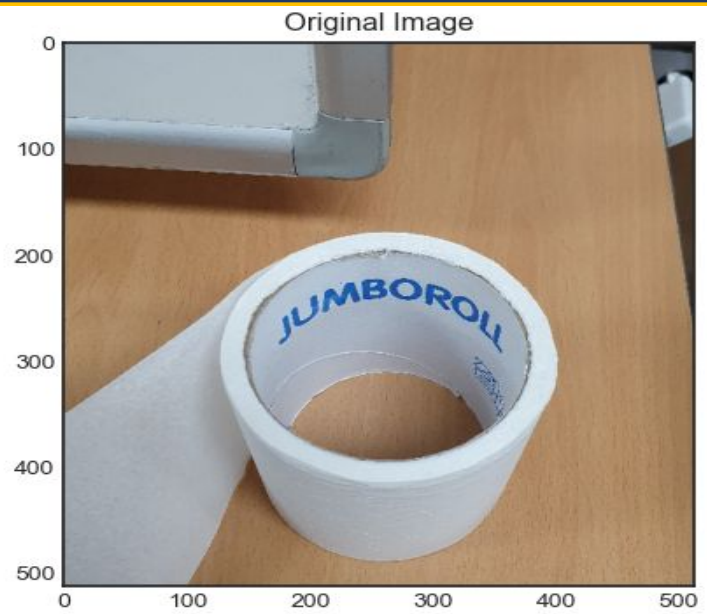


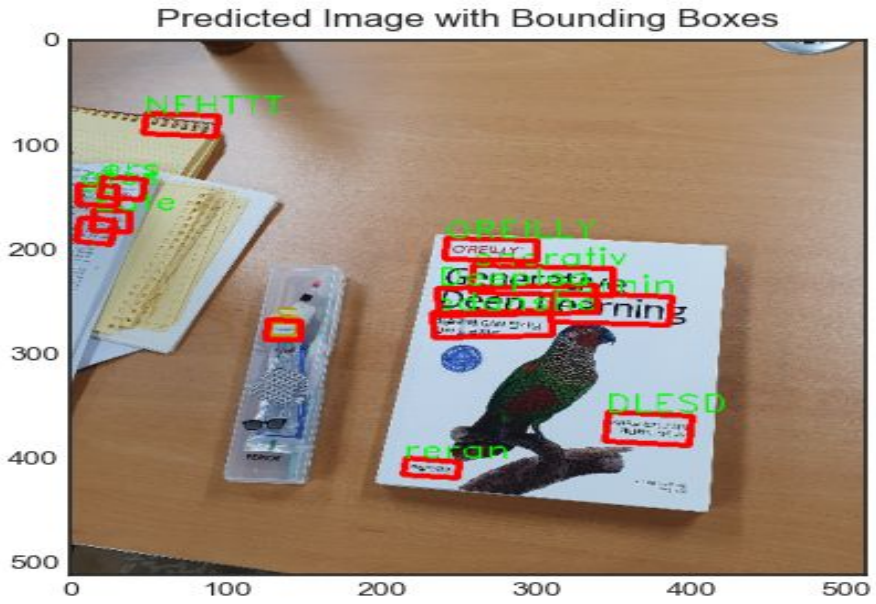




FOTS : Eval Output comparison 2







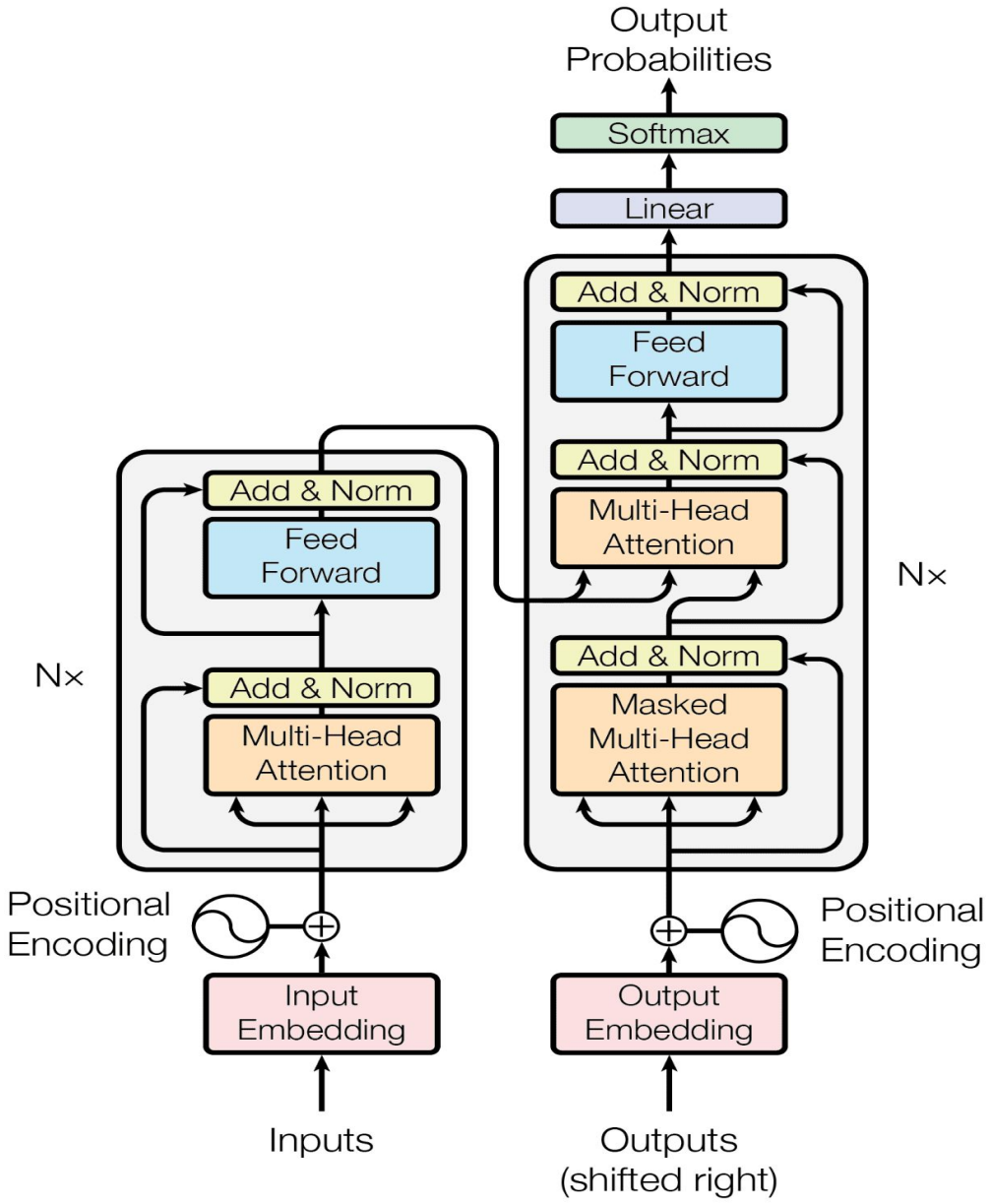
FOTS : Eval Output comparison 2 한계점

QUAD 사각형
quadrilateral
자유도 8 :
(x1,y1,x2,y2,X3,y3,x4,y4)

POLY는 Detecting 제한 됨

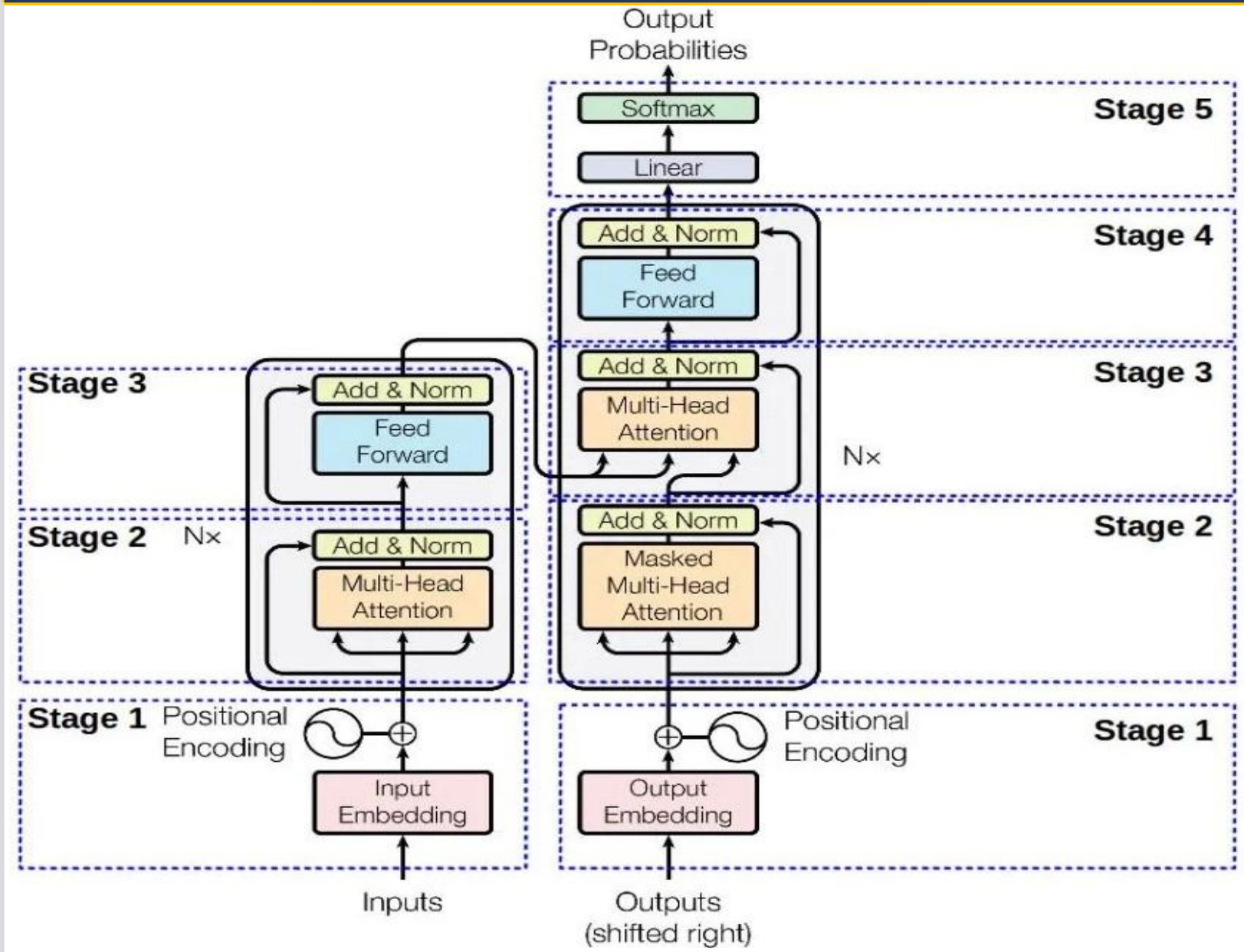
POLY 다각형
Polygon
N개의 점으로 표현 시
자유도 2N

Transformer : architecture

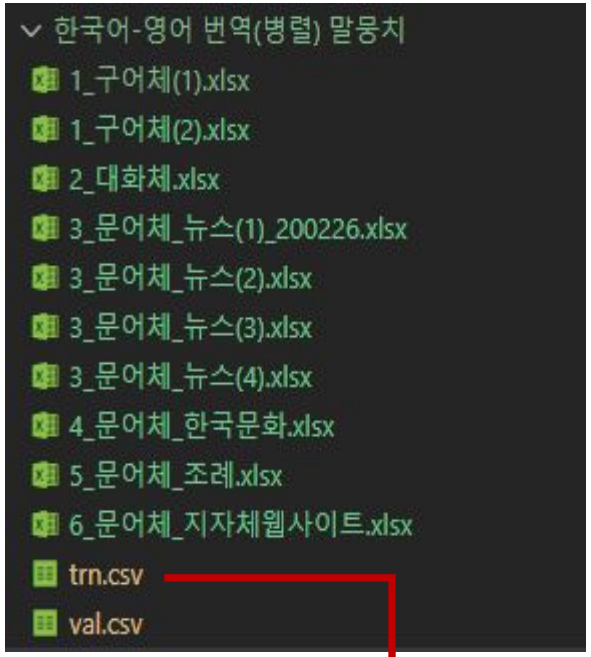


- RNN을 사용하지 않은 병렬 처리 연산으로 계산 속도를 향상 시킴
- 6개의 encoder layer와 6개의 decoder layer로 구성되어있음
- 문맥의 순서정보를 부여해줄 Positional Embedding을 추가

Transformer : architecture



Transformer : dataset



원문,번역문
바위 위에 이름을 적거나 낙서를 하지 마세요.,Do not paint names or graffiti on rocks.
이 내용에 비판적인 댓글이 달리자 작성자는 당일 '장난이었다'는 식으로 다시 댓글을 단 뒤 글을 삭제했다.,"When t
나는 당신에게 음료보다 물을 많이 마실 것을 권해요.,I recommend you to drink water rather than beverages.
다만 카풀이 전업화되는 걸 막기 위해 별도 직업이 있는 경우로 드라이버 자격을 제한하는 조건을 제시했다.,"Howeve
지역 아마추어 야구팀의 위축을 내세우는 것은 명분이 너무 약하다.,It is a weak cause to put forward the cont
그녀는 신비스러운 여자입니다.,She is a mysterious woman.
"또한 산과 바다, 도시와 농지가 복합돼 있는 지역이다.","It is also an area where mountain and sea, and cit
수 많은 사람들의 땀과 노고가 담긴 영화가 너무나도 쉽게 폄하되고 평가절하 되고 있는 작금의 현실이 안타깝고 개탄
삼겹살은 베이컨과는 또 다른 맛과 특색이 있다.,Pork belly tastes differently and has a distinct feature fr
사용 후에는 기초화장품으로 피부를 관리해줍니다.,"After use, use basic cosmetic for skin care."
원래 나의 예약을 그대로 유지하고 싶어요.,I want to keep my original contract.
미국 듀크대 교수가 중국 유학생들에게 교내에서는 영어만 사용해 달라는 내용의 이메일을 보냈다가 학생들의 반발로
많은 선수들이 하루에 다섯 종목씩 치르려면 숨 고를 틈도 없겠군요.,There is no time for getting their breath
경기도와 반려동물 사료회사 6개사가 손을 모아 도내 유기견을 위해 '사랑의 사료'를 전달한다.,"Six companies fro
취업준비를 하며 대학생활을 마무리하는 시기가 되었습니다.,It's now time for me to finish my college life an
경찰 관계자는 "중장비 등으로 토사 제거 후 보강 작업을 벌이고 있으며 이 구간 통행 차량을 부분 통제하고 있다"며
"나는 먼저 갈게, 열심히 걸어서 학교에 와!","I'll go on ahead, have fun walking to school!"
그는 왜 그의 발을 쳐다보냐고 내게 물었다.,He asked me why I'm looking at his feet.

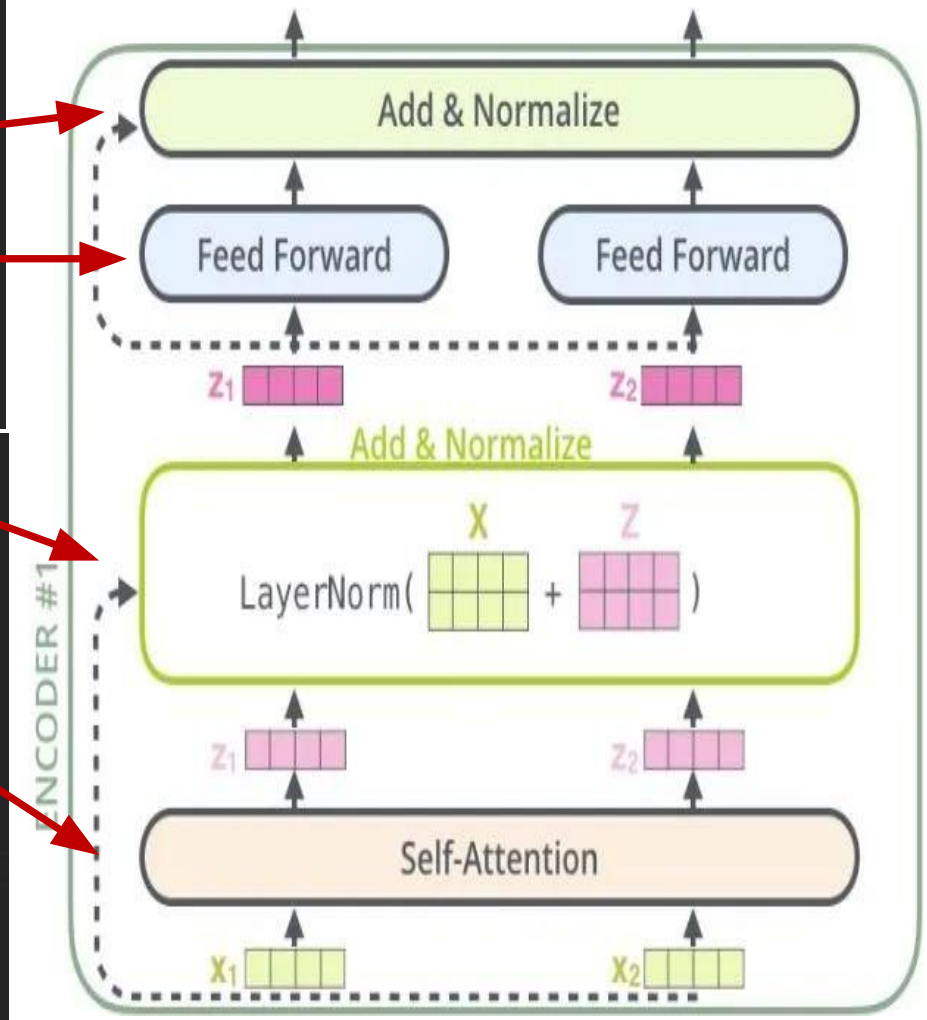
Transformer : Source code

Torch ver.

```
class EncoderLayer(nn.Module):
    def __init__(self, hidden_dim, n_heads, pf_dim, dropout_ratio, device):
        super().__init__()

        self.ff_layer_norm = nn.LayerNorm(hidden_dim)
        self.positionwise_feedforward = PositionwiseFeedforwardLayer(hidden_dim, pf_dim, dropout_ratio)
        self.self_attn_layer_norm = nn.LayerNorm(hidden_dim)
        self.self_attention = MultiHeadAttentionLayer(hidden_dim, n_heads, dropout_ratio, device)
        self.dropout = nn.Dropout(dropout_ratio)
```

Tensor ver.

```
class TransformerEncoder(layers.Layer):
    def __init__(self, embed_dim, dense_dim, num_heads, **kwargs):
        super(TransformerEncoder, self).__init__(**kwargs)
        self.embed_dim = embed_dim
        self.dense_dim = dense_dim
        self.num_heads = num_heads
        self.attention = layers.MultiHeadAttention(
            num_heads=num_heads, key_dim=embed_dim
        )
        self.dense_proj = keras.Sequential(
            [layers.Dense(dense_dim, activation="relu"), layers.Dense(embed_dim),]
        )
        self.layernorm_1 = layers.LayerNormalization()
        self.layernorm_2 = layers.LayerNormalization()
        self.supports_masking = True

    def call(self, inputs, mask=None):
        if mask is not None:
            padding_mask = tf.cast(mask[:, tf.newaxis, tf.newaxis, :], dtype="int32")
        attention_output = self.attention(query=inputs, value=inputs, key=inputs, attention_mask=padding_mask)
        proj_input = self.layernorm_1(inputs + attention_output)
        proj_output = self.dense_proj(proj_input)
        return self.layernorm_2(proj_input + proj_output)
```



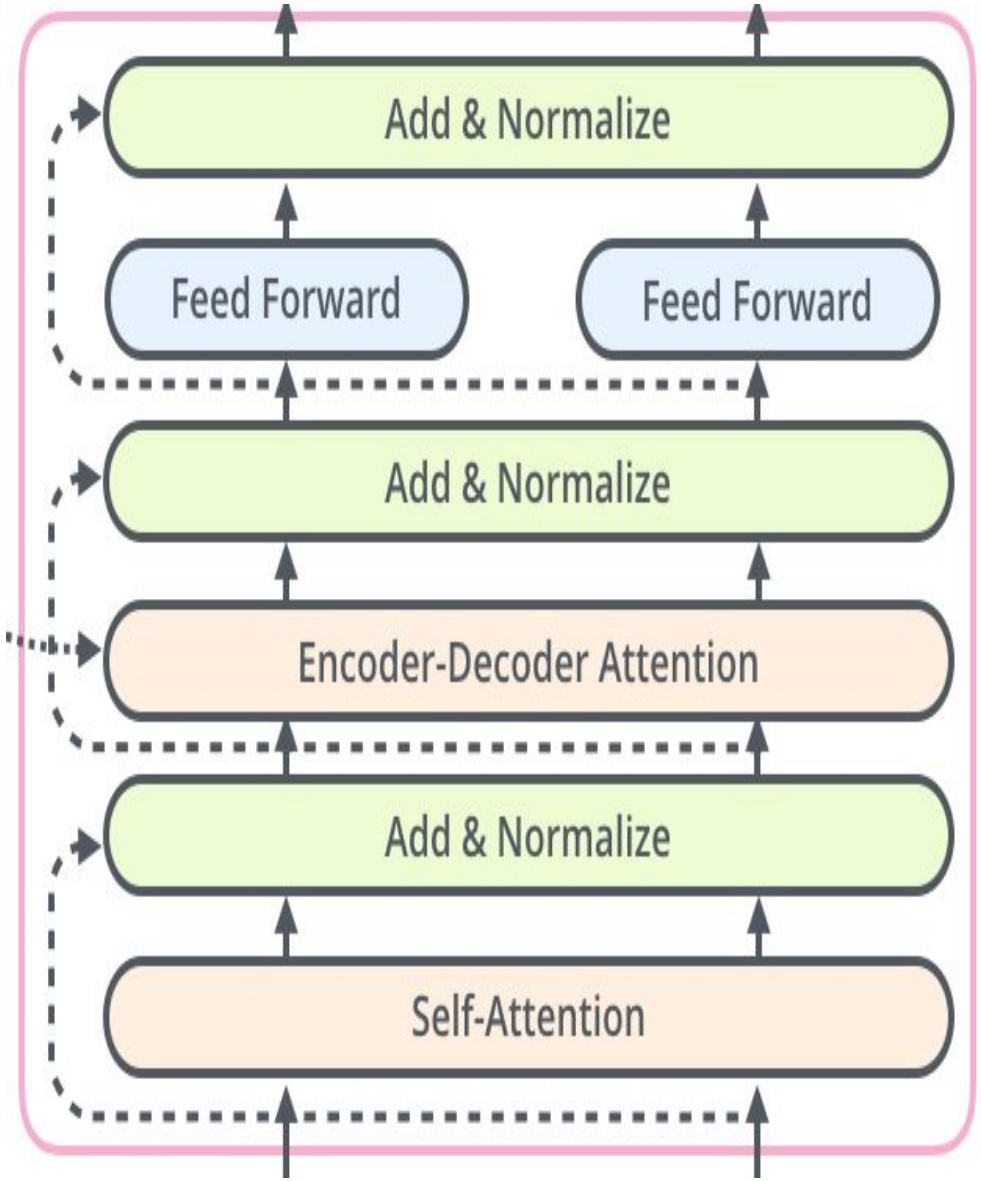
Transformer : Source code

Torch ver.

```
class DecoderLayer(nn.Module):
    def __init__(self, hidden_dim, n_heads, pf_dim, dropout_ratio, device):
        super().__init__()
        self.ff_layer_norm = nn.LayerNorm(hidden_dim)
        self.positionwise_feedforward = PositionwiseFeedforwardLayer(hidden_dim, pf_dim, dropout_ratio)
        self.enc_attn_layer_norm = nn.LayerNorm(hidden_dim)
        self.encoder_attention = MultiHeadAttentionLayer(hidden_dim, n_heads, dropout_ratio, device)
        self.self_attn_layer_norm = nn.LayerNorm(hidden_dim)
        self.self_attention = MultiHeadAttentionLayer(hidden_dim, n_heads, dropout_ratio, device)
        self.dropout = nn.Dropout(dropout_ratio)
```

Tensor ver.

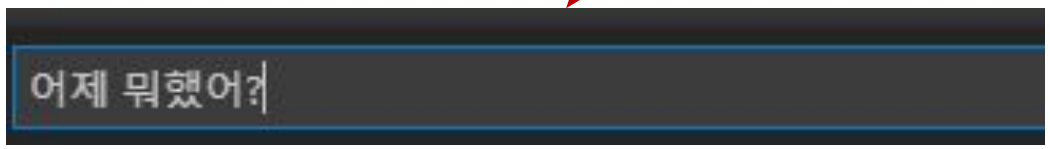
```
class TransformerDecoder(layers.Layer):
    def __init__(self, embed_dim, latent_dim, num_heads, **kwargs):
        super(TransformerDecoder, self).__init__(**kwargs)
        self.embed_dim = embed_dim
        self.latent_dim = latent_dim
        self.num_heads = num_heads
        self.attention_1 = layers.MultiHeadAttention(
            num_heads=num_heads, key_dim=embed_dim
        )
        self.attention_2 = layers.MultiHeadAttention(
            num_heads=num_heads, key_dim=embed_dim
        )
        self.dense_proj = keras.Sequential(
            [layers.Dense(latent_dim, activation="relu"), layers.Dense(embed_dim),]
        )
        self.layernorm_1 = layers.LayerNormalization()
        self.layernorm_2 = layers.LayerNormalization()
        self.layernorm_3 = layers.LayerNormalization()
        self.supports_masking = True
```

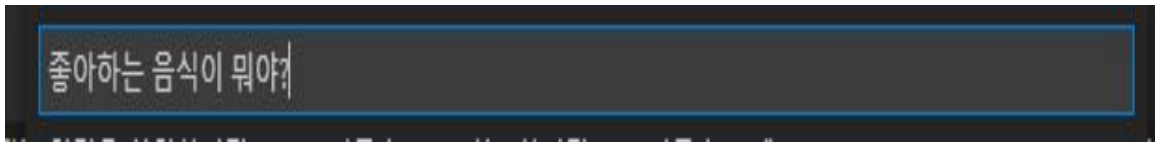


Transformer : Output

```
src = input()

translation, attention = translate_sentence(src, SRC, TRG, model, device, logging=True)

print('')
print("모델 번역 결과:", " ".join(translation))
```

✓ 6.5s

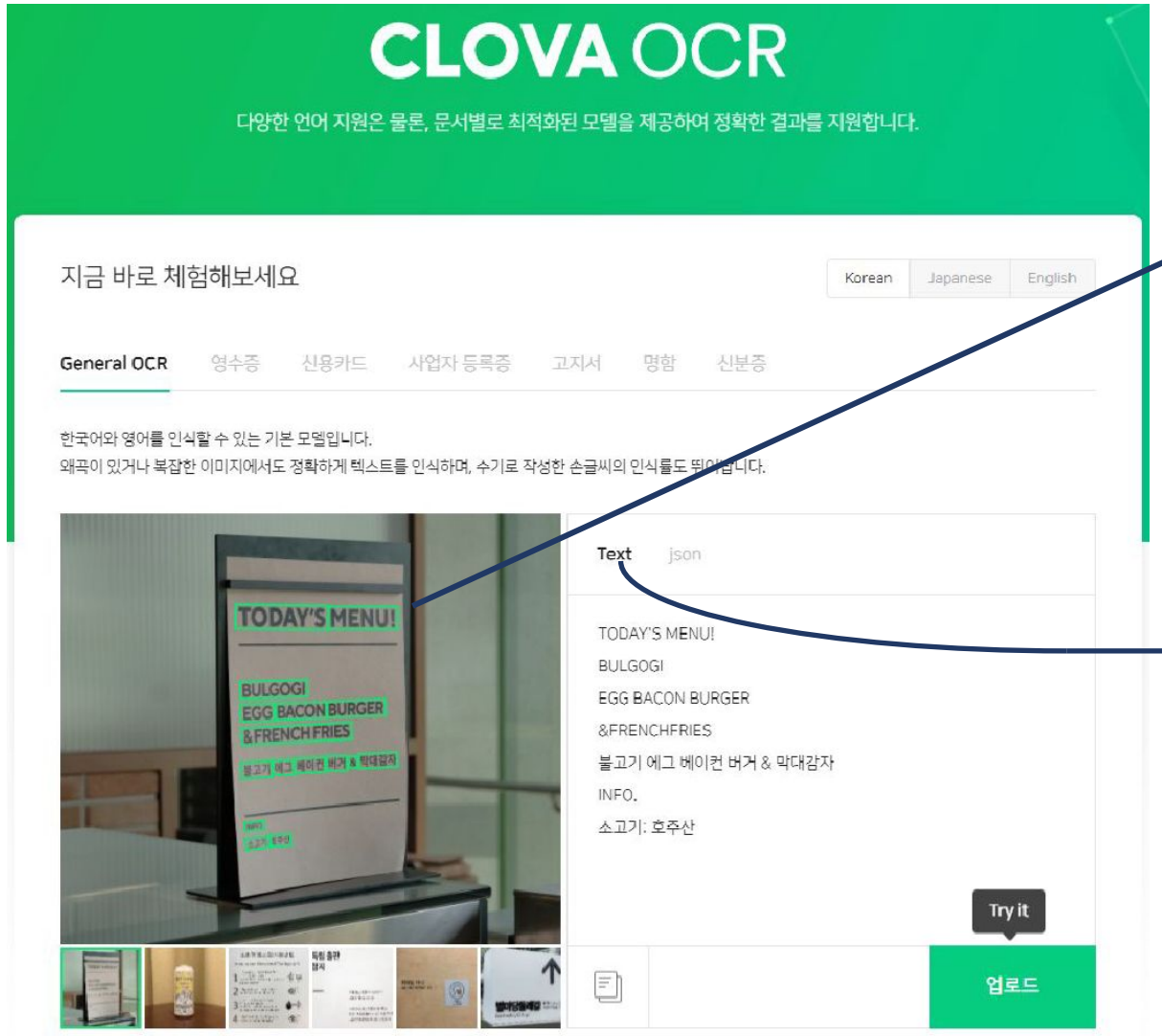




모델 번역 결과: did you haven't yesterday. <eos>

모델 번역 결과: what is your food <unk> <eos>

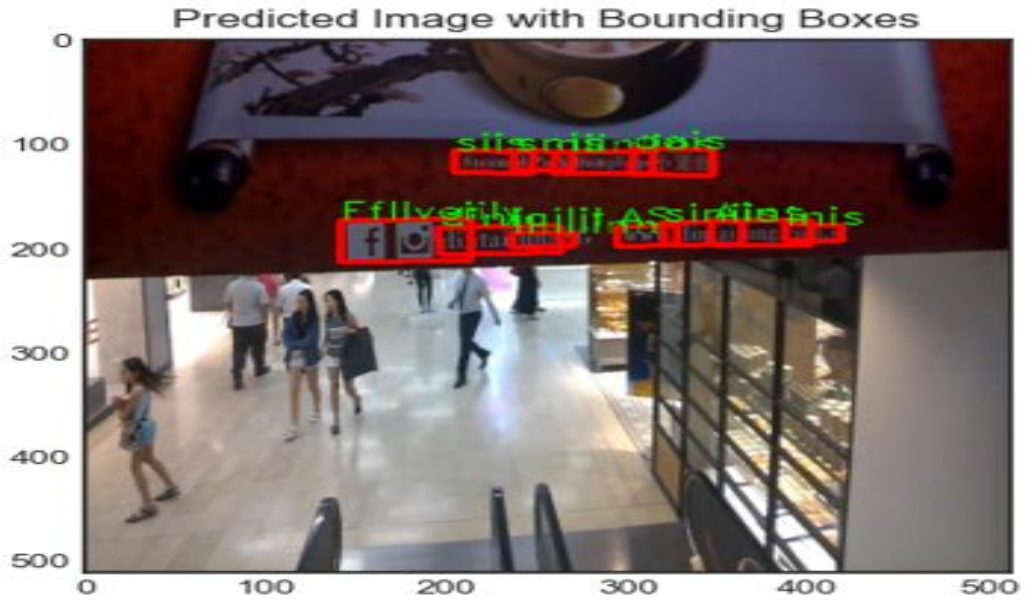
Output



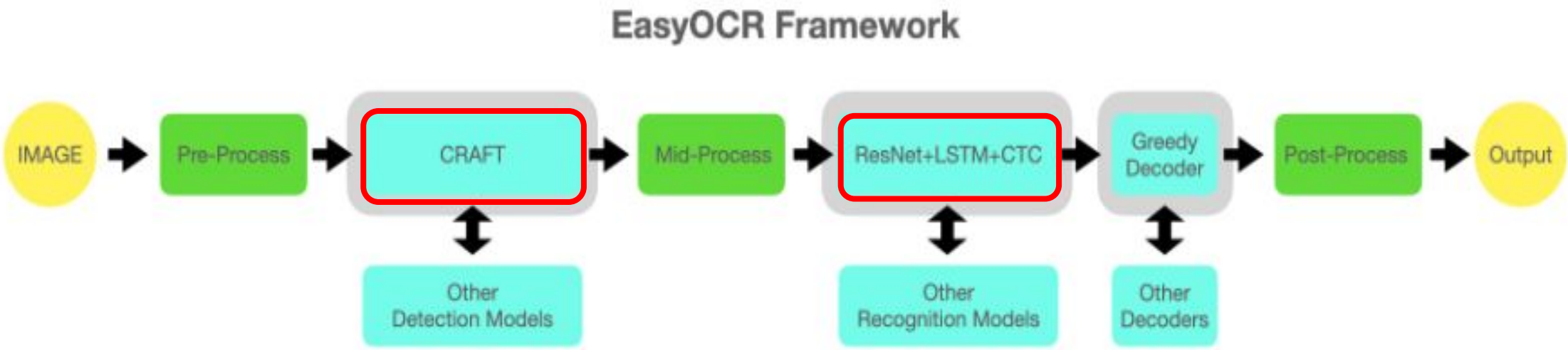
이미지 디텍션/인식

트랜스포머를 이용한 번역

현재 FOTS 모델의 한계점



Prototype：easyOCR Framework



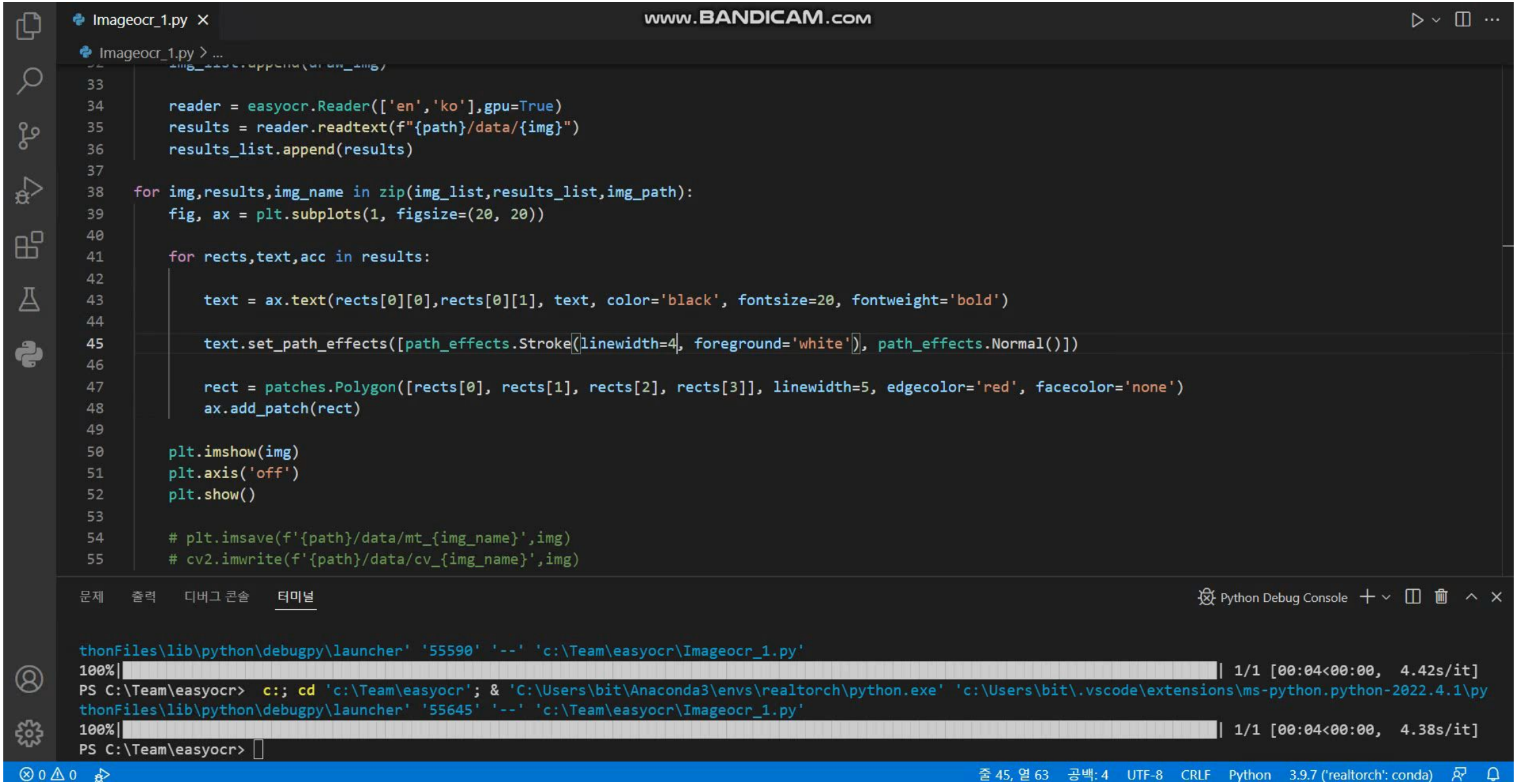
Prototype : easyOCR + Translation

```
for img in tqdm(img_path):  # tqdm 프로그레스 바

    draw_img = cv2.cvtColor(cv2.imread(f"{path}/data/{img}"),cv2.COLOR_BGR2RGB)
    # img = np.array(Image.open(f"{path}/data/{img}").rotate(-90))
    img_list.append(draw_img)

    reader = easyocr.Reader(['en','ko'],gpu=True)
    results = reader.readtext(f"{path}/data/{img}")
    results_list.append(results)

for img,results,img_name in zip(img_list,results_list,img_path):
    fig, ax = plt.subplots(1, figsize=(20, 20))

    for rects,text,acc in results:

        text = ax.text(rects[0][0],rects[0][1], text, color='black', fontsize=20, fontweight='bold')

        text.set_path_effects([path_effects.Stroke(linewidth=4, foreground='white'), path_effects.Normal()])

        rect = patches.Polygon([rects[0], rects[1], rects[2], rects[3]], linewidth=5, edgecolor='red', facecolor='none')
        ax.add_patch(rect)

    plt.imshow(img)
    plt.axis('off')
    plt.show()
```

Prototype : easyOCR + Translation

```
options = webdriver.ChromeOptions()
options.add_experimental_option('excludeSwitches', ['enable-logging'])
driver = webdriver.Chrome('C://chromedriver.exe', options=options)
driver.get("https://translate.google.co.kr/?hl=ko")
driver.maximize_window()  # 창 최대로 키워줌

translate_list = []

for results in results_list:
    tr_list = []

    for rects,text,acc in results:
        elem = driver.find_element_by_xpath('/html/body/c-wiz/div/div[2]/c-wiz/div[2]/c-wiz/div[1]/div[2]/div[3]/c-wiz[1]/span/span/div/textarea')
        elem.send_keys(f'{text}')
        time.sleep(3)
        try:
            driver.find_element_by_xpath('/html/body/c-wiz/div/div[2]/c-wiz/div[2]/c-wiz/div[1]/div[2]/div[3]/c-wiz[1]/div[3]/div/div/div/span').click()
            time.sleep(3)
        except:
            pass
        tr = driver.find_element_by_xpath('/html/body/c-wiz/div/div[2]/c-wiz/div[2]/c-wiz/div[1]/div[2]/div[3]/c-wiz[2]/div[7]/div/div[1]').text
        tr_list.append(tr)
        driver.get("https://translate.google.co.kr/?hl=ko")
        time.sleep(2)

    translate_list.append(tr_list)
    driver.close()
```

Prototype : easyOCR + Translation



Problems / Improvements

Problems

1. FOTS Model 정확도 향상

- 정확도 향상을 위한 절대 시간 필요
- 1주일 동안의 훈련만으로 비약적 향상은 어려움

2. Transformer 번역 정확도

- 한글 데이터의 정확도를 높이는 문제점

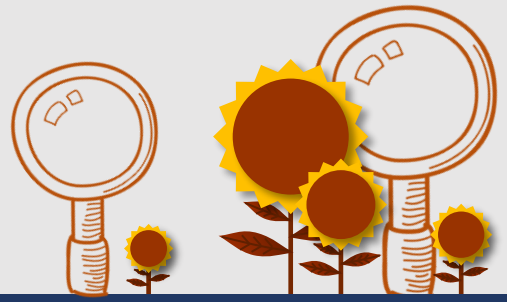
Improvement

1. Flask 구현

- Web으로 구현하기 위한 Flask 학습 필요

2. Real-time 시도

- Fots 모델 활용하여 real-time text 인식 실행



QUESTION