IMAGE CAPTIONING
CAPSULER

강하은 김수민 김수희 당현선 이재준

CONTENTS

01

CAPSULER

프로젝트 개요

이미지를 언어라는 캡슐에 담다.

주제 선정 배경

"시각장애인은 어떻게 인스타를 할까?"

"이미지에 자동으로 캡션을 달 수는 없을까?"

주제 소개

이미지 캡셔닝을 활용한 이미지 리더기 구현

캡션

이미지와 영상의
내용에 맞는
문장을 생성합니다.

키워드

생성된 문장의
키워드만
출력합니다.

번역

생성된 문장을
4개 국어로
번역합니다.

음성

생성된 문장과
번역된 문장을
읽어줍니다.

데이터 소개

이미지 12만장

Train : 82,783

Validation : 40,000

영어 캡션 60만건

한국어 캡션 60만건

https://aihub.or.kr/keti_data_board/visual_intelligence (한국어)

https://cocodataset.org/#home (원어)

데이터 소개

5 english captions per image

translated

5 korean captions per image

image path

```
json_data[0]

{'file_path': 'train2014/COCO_train2014_000000057870.jp
  'captions': ['A restaurant has modern wooden tables an
   'A long restaurant table with rattan rounded back cha
   'a long table with a plant on top of it surrounded wi
   'A long table with a flower arrangement in the middle
   'A table is adorned with wooden chairs with blue acce
  'id': 57870,
  'caption_ko': ['식당에는 현대식 나무 탁자와 의자가 있다.',
   '등이 둥근 라탄 모양으로 된 긴 레스토랑 탁자가 의자에 앉아 있다.',
   '나무 의자로 둘러싸인 식물이 위에 있는 긴 탁자',
   '중앙에 회의를 위해 꽃을 배열한 긴 테이블',
   '테이블이 푸른 색 악센트가 있는 나무 의자로 장식되어 있다.']}
```

```
Image.open(json_filepath[0])
```

https://aihub.or.kr/keti_data_board/visual_intelligence (한국어)
https://cocodataset.org/#home (원어)

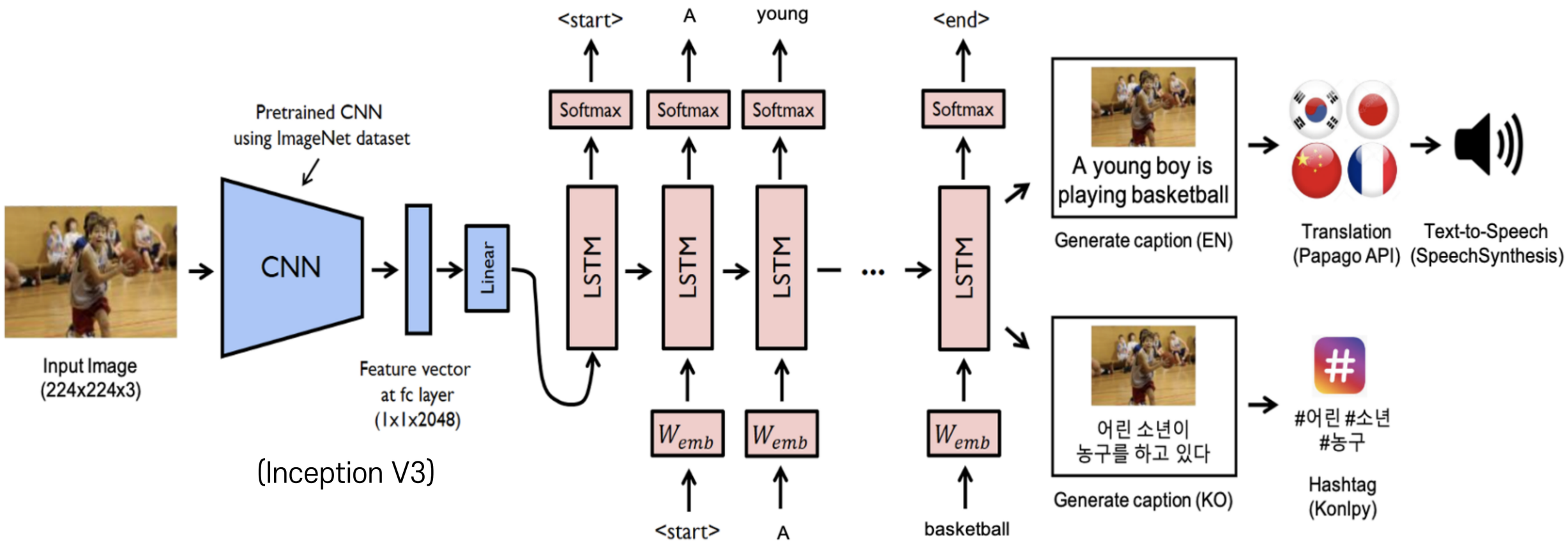
IMAGE CAPTIONING : What is it?

'네 명의 사람들이 잔디 위에서 뛰고 있다.'

'Four people are jumping on the grass.'

IMAGE CAPTIONING

OPENCV / CNN + RNN / TRANSLATION / SPEECH SYNTHESIS / TOKENIZING



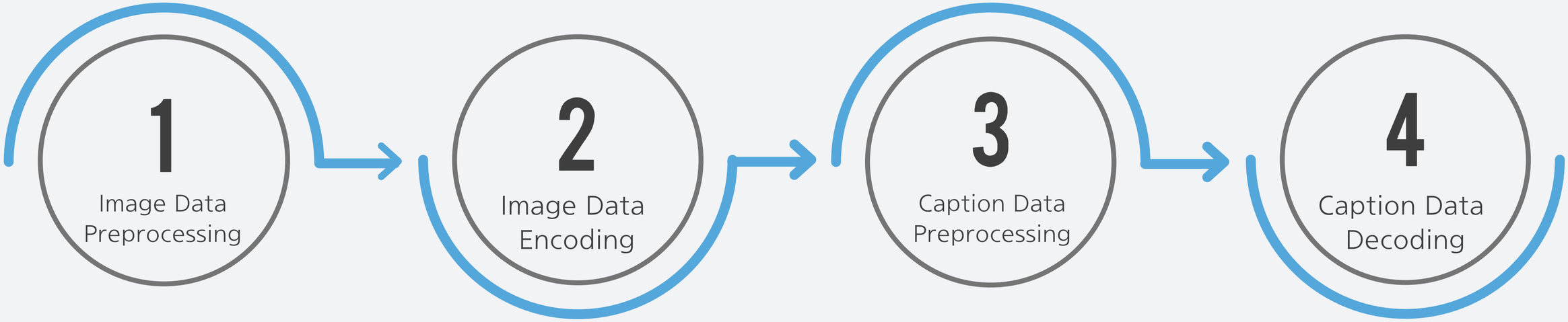
02

CAPSULER

모델 소개 & 코드리뷰

이미지를 언어라는 캡슐에 담다.

DATA (PRE)PROCESSING



MODEL ARCHITECTURE 1 : CNN

Problems of CNN:

MODEL ARCHITECTURE 1 : CNN

Solution: GoogLeNet's Inception Module

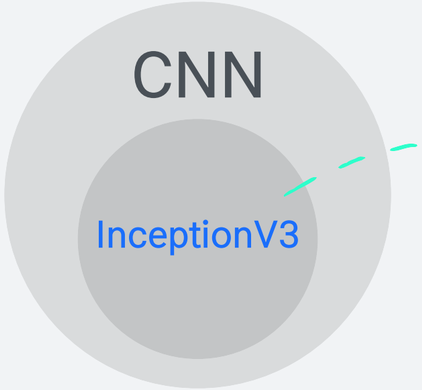
정확도

Simultaneous Filters

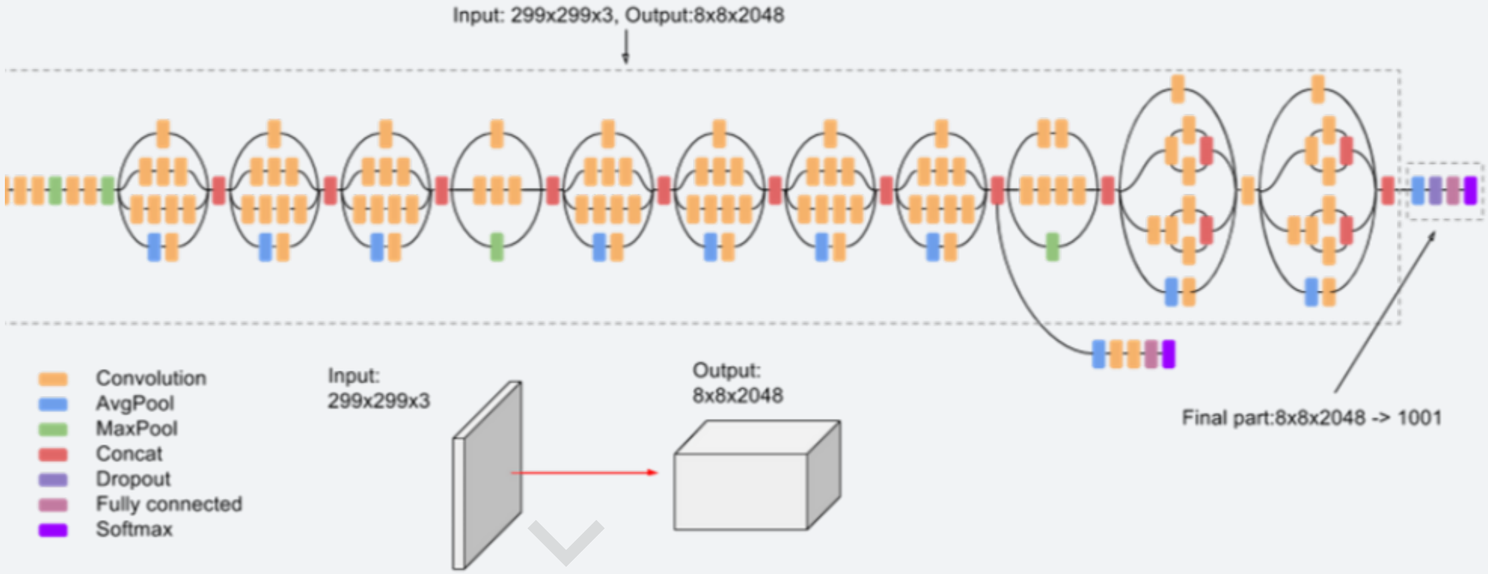
Efficiency

https://ikkison.tistory.com/86

MODEL ARCHITECTURE 1 : CNN



Google Research 에서 개발한 CNN 모델



DATA PREPROCESING : IMAGE

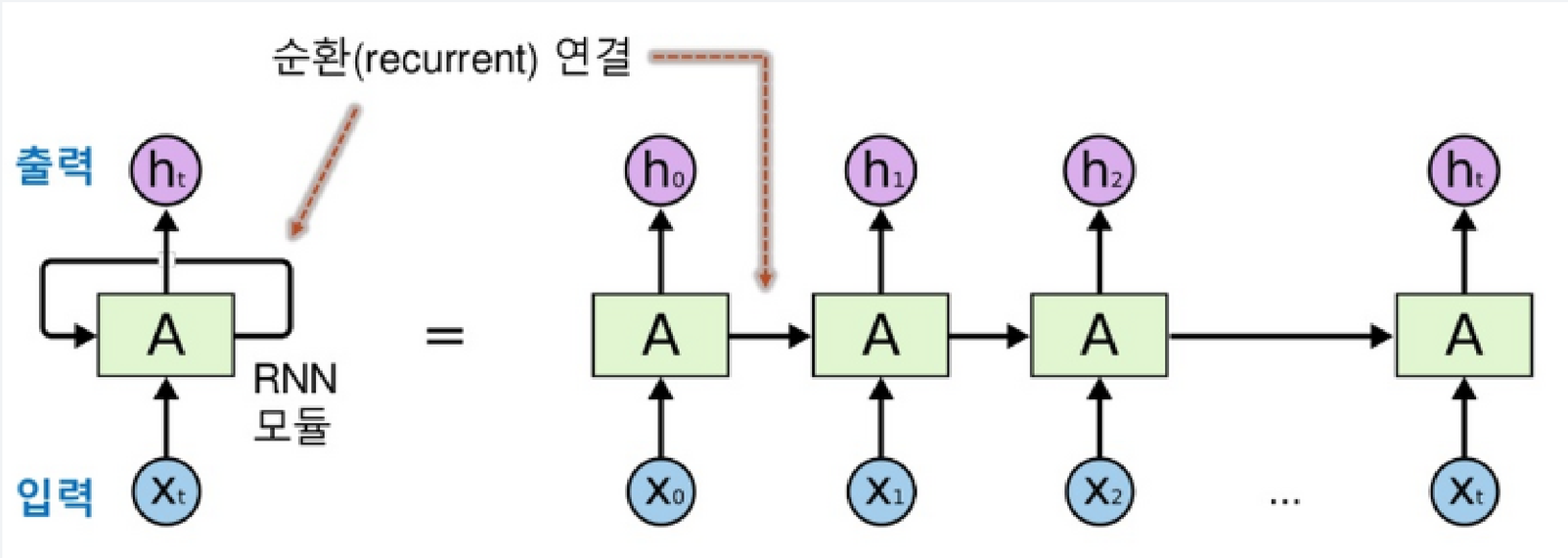
```python
encode_model = InceptionV3(weights='imagenet')
encode_model = Model(encode_model.input, encode_model.layers[-2].output)

def encodeImage(img):
  img = img.resize((299, 299), Image.ANTIALIAS)
  x = tensorflow.keras.preprocessing.image.img_to_array(img)
  x = np.expand_dims(x, axis=0)
  x = tensorflow.keras.applications.inception_v3.preprocess_input(x)
  x = encode_model.predict(x)
  x = np.reshape(x, 2048)
  return x
```

"Automatic Feature Engineering"

각 이미지 -> 고정된 사이즈의 벡터값으로 전처리

MODEL ARCHITECTURE 2: RNN

Recurrent Neural Network: 순환구조
음성, 문장 등 순차성 지닌 데이터에 적합



https://brunch.co.kr/@linecard/324

MODEL ARCHITECTURE 2: RNN

Problems of RNN:

https://brunch.co.kr/@linecard/324

MODEL ARCHITECTURE 2: RNN

Problems of RNN:

"I am from Korea, and I speak _____."

" Korean "

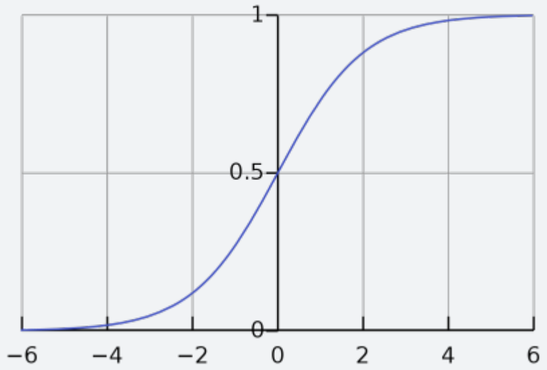
"I am from Korea, and my hometown is Suwon. My hobbies are reading, solving puzzles, playing mobile games, and working out. I usually sleep 6 hours a day. Let's be friends! Oh, and I speak _____."

" ?? "

https://brunch.co.kr/@linecard/324

MODEL ARCHITECTURE 2: RNN

Solution: LSTM (Long-Short term memory networks)

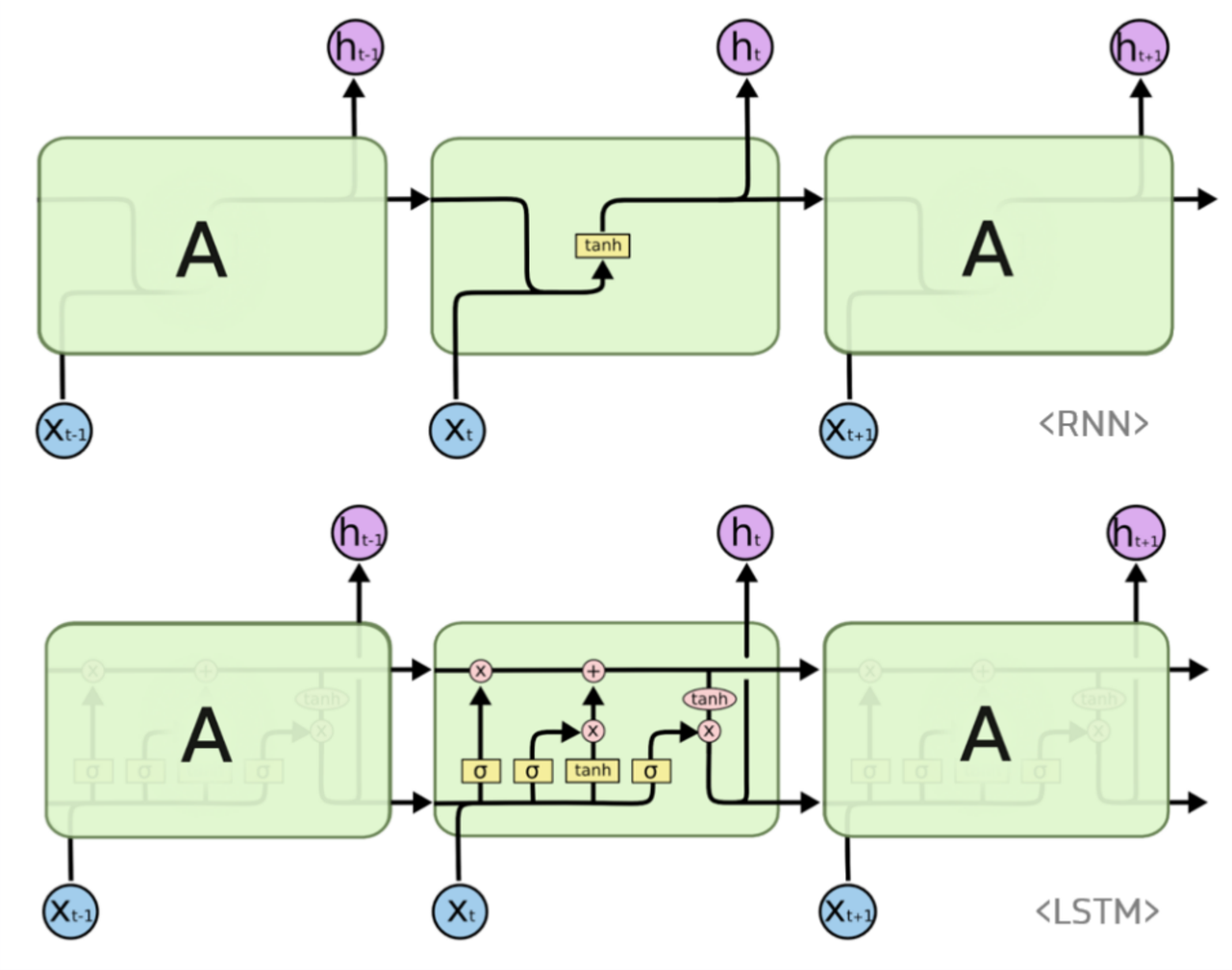


Sigmoid Layers

0 = 데이터 제거
1 = 데이터 유지

https://brunch.co.kr/@linecard/324

MODEL ARCHITECTURE 2: RNN



DATA PREPROCESING : CAPTIONS

1 PADDING

2 SEQUENCING

3 WORD EMBEDDING

https://nlp.stanford.edu/projects/glove/ (GloVe)
https://github.com/Kyubyong/wordvectors.(W2V, FastText)

DATA PREPROCESSING : CAPTIONS

PADDING : 전체 문장의 길이를 동일하게 맞춰주는 작업

sentence = ['I love python', 'He loves java', 'We all are amazing software engineers']

sentence = [['I', 'love', 'python'], ['He', 'loves', 'java'], ['We', 'all', 'are', 'amazing', 'software', 'engineers']]

max length = 6

Padding

['I', 'love', 'python', 0, 0, 0]
['He', 'loves', 'java', 0, 0, 0]
['We', 'all', 'are', 'amazing', 'software', 'engineers']

DATA PREPROCESSING : CAPTIONS

PADDING

: 전체 문장의 길이를 동일하게 맞춰주는 작업

```
all_train_captions = [] #모든 캡션을 모은 리스트
null_punct = str.maketrans('', '', string.punctuation)
max_length = 0 #캡션 중 가장 긴 길이

for path, captions in train_descriptions.items():
    caption_list = []
    for caption in captions: # 캡션 간단한 전처리
      caption = caption.split()
      caption = [word.lower() for word in caption]
      caption = [w.translate(null_punct) for w in caption]
      caption = [word for word in caption if len(word)>1]
      caption = [word for word in caption if word.isalpha()]

      max_length = max(max_length, len(caption))

      caption = ' '.join(caption)
      caption = f'STARTSEQ {caption} STOPSEQ' # 캡션 예측시, 예측 시작과 정지를 위한 단어

      all_train_captions.append(caption)
      caption_list.append(caption)

    train_descriptions[path] = caption_list
max_length +=2
print(max_length)
```

```
49
```

Max Length

DATA PREPROCESING : CAPTIONS

1 PADDING

2 SEQUENCING

3 WORD EMBEDDING

https://nlp.stanford.edu/projects/glove/ (GloVe)
https://github.com/Kyubyong/wordvectors.(W2V, FastText)

DATA PREPROCESSING : CAPTIONS

SEQUENCING : 문장에서 단어들을 추출해 순서를 부여하는 작업

```
all_train_captions = [] #모든 캡션을 모은 리스트
null_punct = str.maketrans('', '', string.punctuation)
max_length = 0 #캡션 중 가장 긴 길이

for path, captions in train_descriptions.items():
    caption_list = []
    for caption in captions: # 캡션 간단한 전처리
      caption = caption.split()
      caption = [word.lower() for word in caption]
      caption = [w.translate(null_punct) for w in caption]
      caption = [word for word in caption if len(word)>1]
      caption = [word for word in caption if word.isalpha()]

      max_length = max(max_length, len(caption))

      caption = ' '.join(caption)
      caption = f'STARTSEQ {caption} STOPSEQ' # 캡션 예측시, 예측 시작과 정지를 위한 단어

      all_train_captions.append(caption)
      caption_list.append(caption)

    train_descriptions[path] = caption_list
max_length +=2
print(max_length)
```

```
49
```

STARTSEQ
+
CAPTION
+
STOPSEQ

DATA PREPROCESSING : CAPTIONS

SEQUENCING : 문장에서 단어들을 추출해 순서를 부여하는 작업

Ko.

```
('train2014/COCO_train2014_000000308641.jpg',
 ['주차 미터기 2개에 파란 색 커버가 덮여 있습니다.',
  '주차 미터기에는 그 위에 가방이 있다.',
  '두대의 주차 미터기가 있지만 하나만 사용 중입니다.',
  '1미터가 파손된 이중 주차 미터기',
  '커버가 달린 주차 미터기 2개'])
```

STARTSEQ + CAPTION + STOPSEQ

```
('train2014/COCO_train2014_000000308641.jpg',
 ['startseq 주차 미터기 2개에 파란 색 커버가 덮여 있습니다. endseq',
  'startseq 주차 미터기에는 그 위에 가방이 있다. endseq',
  'startseq 두대의 주차 미터기가 있지만 하나만 사용 중입니다. endseq',
  'startseq 1미터가 파손된 이중 주차 미터기 endseq',
  'startseq 커버가 달린 주차 미터기 2개 endseq'])
```

Eng.

```
('train2014/COCO_train2014_000000057870.jpg',
 ['A restaurant has modern wooden tables and chairs.',
  'A long restaurant table with rattan rounded back chairs.',
  'a long table with a plant on top of it surrounded with wooden chairs ',
  'A long table with a flower arrangement in the middle for meetings',
  'A table is adorned with wooden chairs with blue accents.'])
```

STARTSEQ + CAPTION + STOPSEQ

```
('train2014/COCO_train2014_000000057870.jpg',
 ['STARTSEQ restaurant has modern wooden tables and chairs STOPSEQ',
  'STARTSEQ long restaurant table with rattan rounded back chairs STOPSEQ',
  'STARTSEQ long table with plant on top of it surrounded with wooden chairs STOPSEQ',
  'STARTSEQ long table with flower arrangement in the middle for meetings STOPSEQ',
  'STARTSEQ table is adorned with wooden chairs with blue accents STOPSEQ'])
```

DATA PREPROCESSING : CAPTIONS

SEQUENCING

: 문장에서 단어들을 추출해 순서를 부여하는 작업

```
def data_generator(descriptions, photos, wordtoidx, \
                   max_length, num_photos_per_batch):
  # x1 - Training data for photos
  # x2 - The caption that goes with each photo
  # y - The predicted rest of the caption
  x1, x2, y = [], [], []
  n=0
  while True:
    for key, desc_list in descriptions.items(): #train_description( {'image path
      n+=1
      photo = photos[key] #photo == image path
      # Each photo has 5 descriptions
      for desc in desc_list:
        # Convert each word into a list of sequences.
        seq = [wordtoidx[word] for word in desc.split(' ') if word in wordtoidx]

        for i in range(1, len(seq)):
          #in_seq => [9],[9,10],[9,10,23],[9,10,23,4]
          #out_seq => [10],[23],..,[5]
          in_seq, out_seq = seq[:i], seq[i]
          in_seq = pad_sequences([in_seq], maxlen=max_length)[0] #max capation 길
          out_seq = to_categorical([out_seq], num_classes=vocab_size)[0] # out_s
          x1.append(photo) #image path
          x2.append(in_seq) #padded caption value
          y.append(out_seq) #one hot encoded caption target
      if n==num_photos_per_batch: #batch size마다 x1,x2,y가 저장된다.
        yield ([np.array(x1), np.array(x2)], np.array(y))
        x1, x2, y = [], [], []
        n=0
```

Example Caption: 'STARTSEQ I love python ENDSEQ'

STARTSEQ (predict: I)
STARTSEQ I (predict: love)
STARTSEQ I love (predict: python)
STARTSEQ I love python (predict: ENDSEQ)
STARTSEQ I love python ENDSEQ

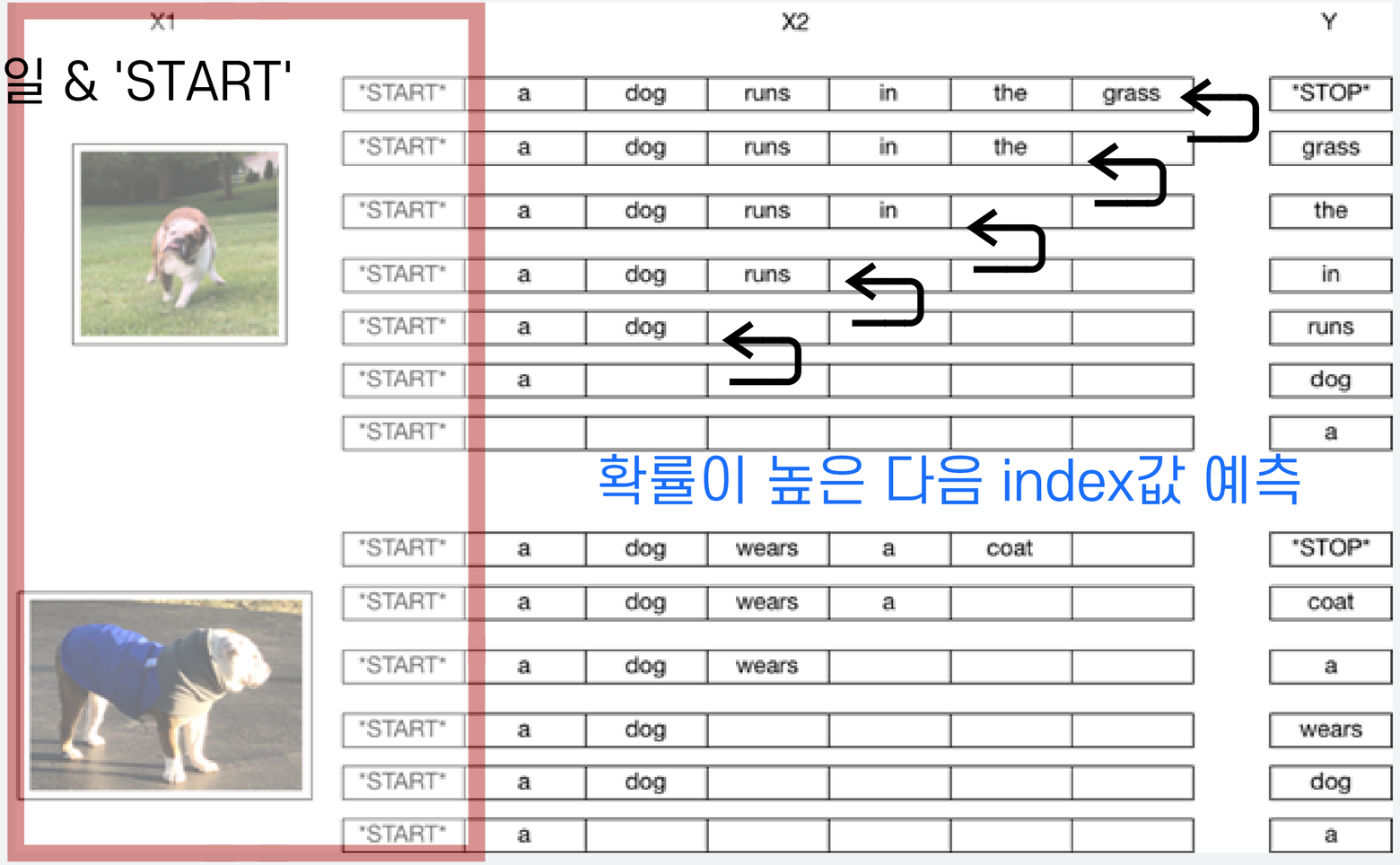
* 함수 안에서 yield를 사용하면 함수는 제너레이터가 됨
* 데이터가 아닌 주소값을 저장하여 메모리 최소화에 유용
* 라이브러리 구성 시 사용빈도 높음

DATA PREPROCESSING : CAPTIONS

SEQUENCING : 문장에서 단어들을 추출해 순서를 부여하는 작업

INPUT : 이미지 파일 & 'START'

'STOP': 예측 정지



DATA PREPROCESING : CAPTIONS

1 PADDING

2 SEQUENCING

3 WORD EMBEDDING

https://nlp.stanford.edu/projects/glove/ (GloVe)
https://github.com/Kyubyong/wordvectors.(W2V, FastText)

DATA PREPROCESING : CAPTIONS

WORD EMBEDDING : 단어의 의미를 밀집 벡터(Dense Vector)로 표현

GloVe

pre-trained vectors

FastText
(ko)

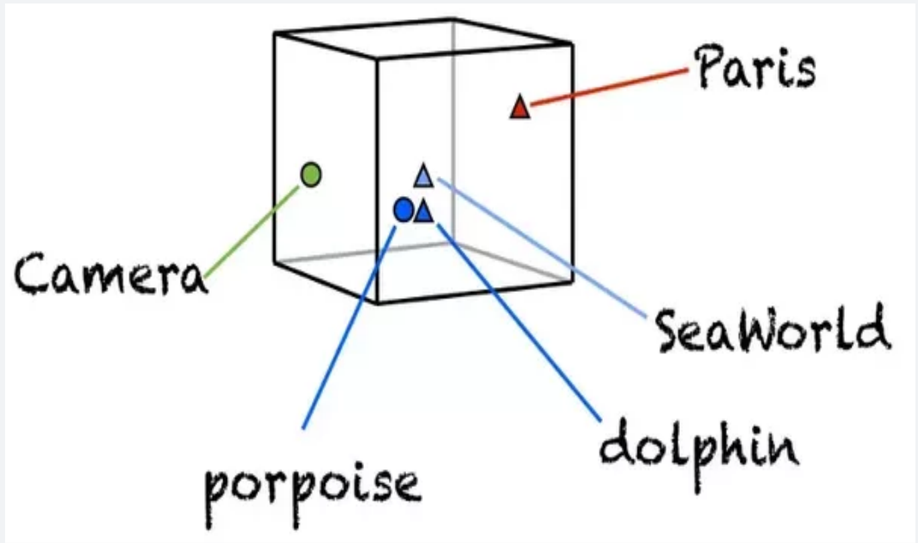
40,000 word vectors
embedding_metrics.shape (6291, 200)

https://nlp.stanford.edu/projects/glove/ (GloVe)
https://github.com/Kyubyong/wordvectors.(W2V, FastText)

DATA PREPROCESING : CAPTIONS

WORD EMBEDDING : 단어의 의미를 밀집 벡터(Dense Vector)로 표현

```
('disability',
 array([-3.0823e-02, -8.2912e-01, -8.2995e-01, -1.6071e-01,  4.6970e-01,
        -5.0182e-01, -2.6778e-01,  6.3755e-01, -3.4764e-01,  7.4109e-02,
         7.2530e-02,  5.1631e-01,  4.6328e-01,  1.4684e-01,  4.8146e-01,
         3.3465e-01,  2.9641e-01,  1.6992e-01,  4.2015e-02,  6.0510e-01,
         6.2413e-01,  1.6726e+00, -4.5826e-01, -1.6596e-01,  1.4549e-01,
         4.4906e-01,  6.0129e-01, -9.3747e-02,  2.0329e-01,  2.9914e-01,
         2.2476e-01, -1.1174e-01, -3.9450e-01,  4.1072e-01,  1.0779e-01,
         2.0593e-01,  3.0417e-01, -5.0885e-01, -2.9464e-01,  6.3562e-01,
         3.2261e-01, -2.4777e-01, -7.1790e-01,  3.5287e-01,  1.1077e-01,
        -3.7492e-01, -2.3488e-01, -7.1785e-02, -6.8215e-02, -5.0620e-01,
```



https://nlp.stanford.edu/projects/glove/ (GloVe)
https://github.com/Kyubyong/wordvectors.(W2V, FastText)

DATA PREPROCESING : CAPTIONS

WORD EMBEDDING : 단어의 의미를 밀집 벡터(Dense Vector)로 표현

GloVe

pre-trained vectors

FastText (ko)

```
ko_model.wv.most_similar(positive=['헬멧을', '오토바이'], negative=['남자'])
```

```
[('유언비어', 0.3286445140838623),
 ('제트기', 0.27454566955566406),
 ('이륜', 0.255718469619751),
 ('변속', 0.24294236302375793),
 ('교자', 0.2403179109096527),
 ('수집', 0.2392008751630783),
 ('식별', 0.23755601048469543),
 ('배달', 0.23647530376911163),
```

https://nlp.stanford.edu/projects/glove/ (GloVe)
https://github.com/Kyubyong/wordvectors.(W2V, FastText)

DATA PREPROCESING : CAPTIONS

WORD EMBEDDING : 단어의 의미를 밀집 벡터(Dense Vector)로 표현

```
embeddings_index = {}
f = open(os.path.join('glove.6B.200d.txt'), encoding="utf-8")

for line in tqdm(f):
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs

f.close()
print(f'Found {len(embeddings_index)} word vectors.')
```

```
embedding_dim = 200

# Get 200-dim dense vector for each of the 10000 words in out vocabulary
embedding_matrix = np.zeros((vocab_size, embedding_dim))

for word, i in wordtoidx.items():
    #if i < max_words:
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        # Words not found in the embedding index will be all zeros
        embedding_matrix[i] = embedding_vector
```

https://nlp.stanford.edu/projects/glove/ (GloVe)
https://github.com/Kyubyong/wordvectors.(W2V, FastText)

MODEL TRAINING

MODEL STRUCTURE

CNN

```
encode_model = InceptionV3(weights='imagenet')
encode_model = Model(encode_model.input, encode_model.layers[-2].output)
```

RNN

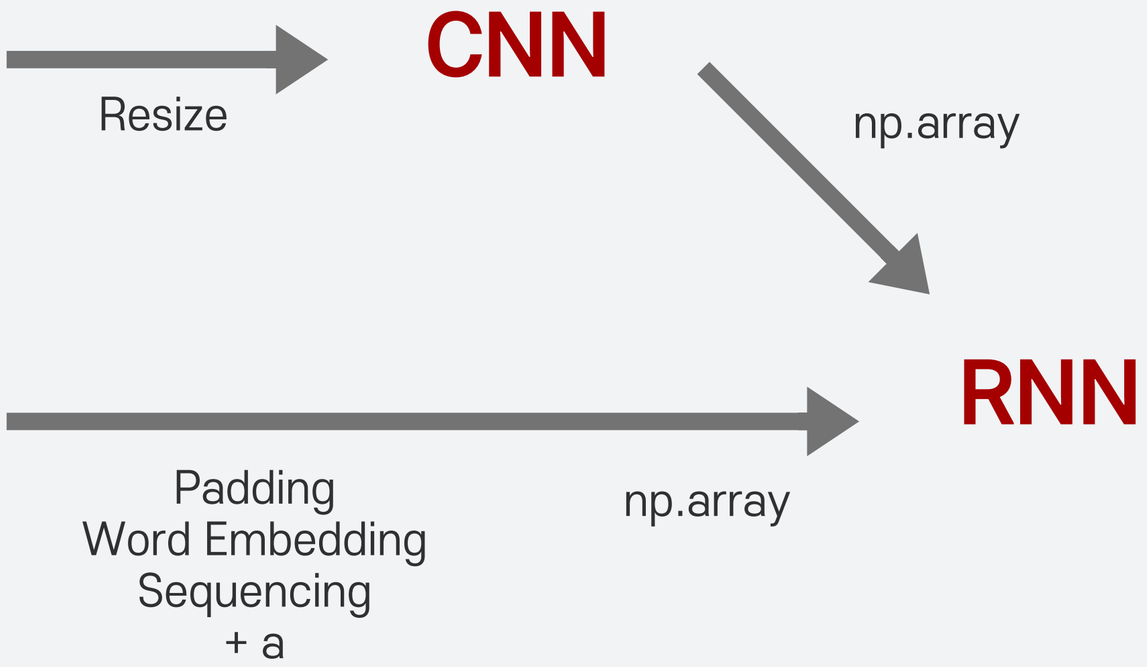
```
inputs1 = Input(shape=(2048,))
fe1 = Dropout(0.5)(inputs1)
fe2 = Dense(256, activation='relu')(fe1)
inputs2 = Input(shape=(max_length,))
se1 = Embedding(vocab_size, embedding_dim, mask_zero=True)(inputs2)
se2 = Dropout(0.5)(se1)
se3 = LSTM(256)(se2)
decoder1 = add([fe2, se3])
decoder2 = Dense(256, activation='relu')(decoder1)
outputs = Dense(vocab_size, activation='softmax')(decoder2)
caption_model = Model(inputs=[inputs1, inputs2], outputs=outputs)
```

```
caption_model.layers[2].set_weights([embedding_matrix])
caption_model.layers[2].trainable = False
caption_model.compile(loss='categorical_crossentropy', optimizer='adam')
```

MODEL TRAINING

MODEL STRUCTURE

'The woman in the kitchen is holding a huge pan.',
'A chef carrying a large pan inside of a kitchen.',
'A woman is holding a large pan in a kitchen. ',
'A woman cooking in a kitchen with granite counters.',
'A woman cooking in her kitchen with a black pan.'



MODEL TRAINING

HYPERPARAMETER TUNING

학습 횟수 늘리기

2만장→4만장→8만장

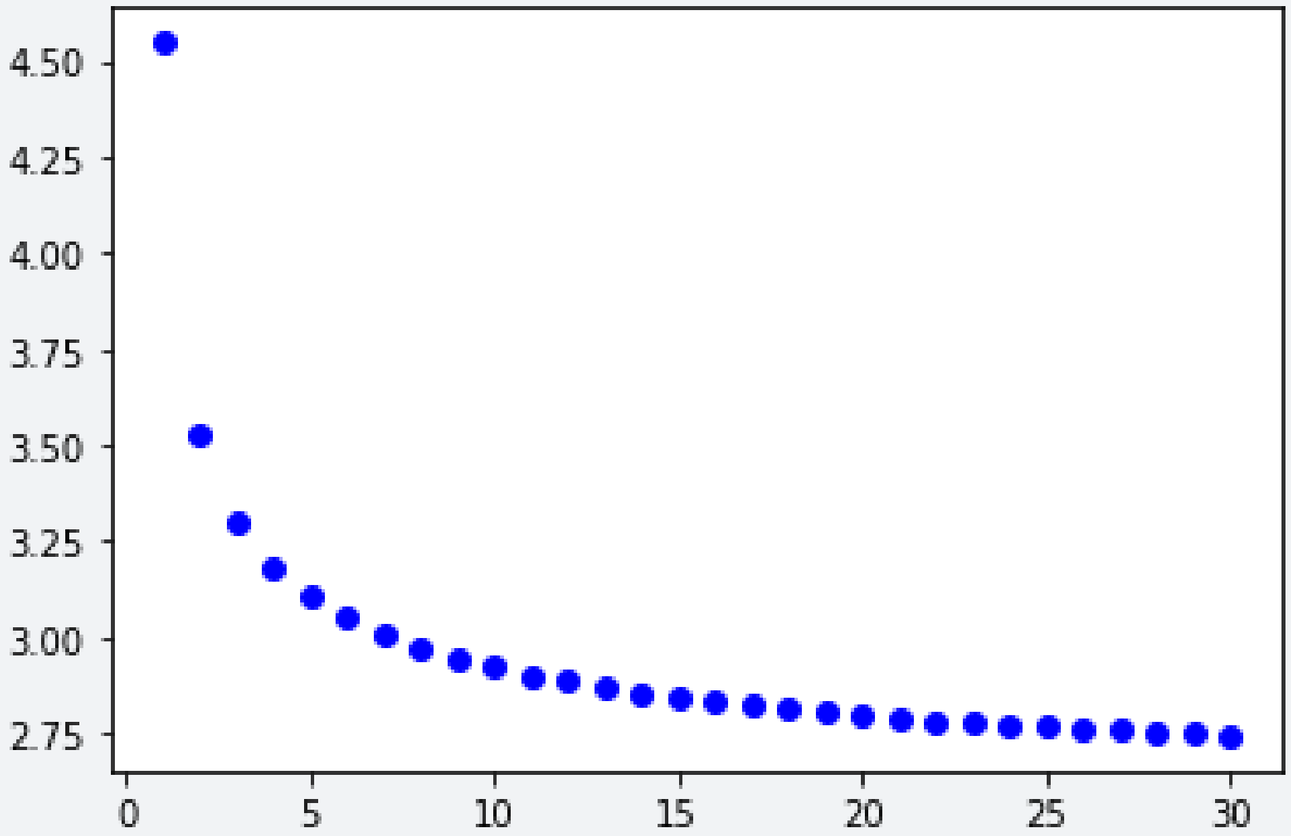
```
# model training
EPOCH = 10
for i in tqdm(range(EPOCH)):
    generator = data_generator(train_descriptions, encoding_train,
                               wordtoidx, max_length, number_pics_per_batch)
    history = caption_model.fit_generator(generator, epochs=1, callbacks=[callback_checkpoint],
                                          steps_per_epoch=steps, verbose=1)
```

배치 사이즈

3→50→300→200

MODEL TRAINING

LOSS RATE



최종 결과

데이터 : 82,000

EPOCH : 30 (+10)

BATCH : 200

한국어 캡션 VS 영어 캡션

한국어
Caption

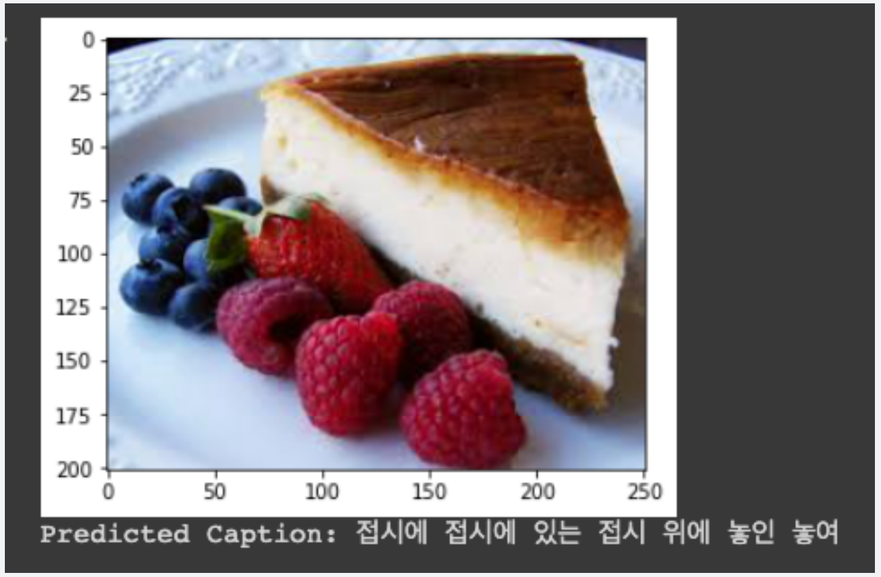
영어
Caption

◎ 이미지 인지 능력
◎ 표현의 다양성

한국어 캡션 VS 영어 캡션

한국어
Caption

영어
Caption



◎ 단어 전처리 부족
◎ 부족한 문장 표현 능력

RESULTS

영어 캡션

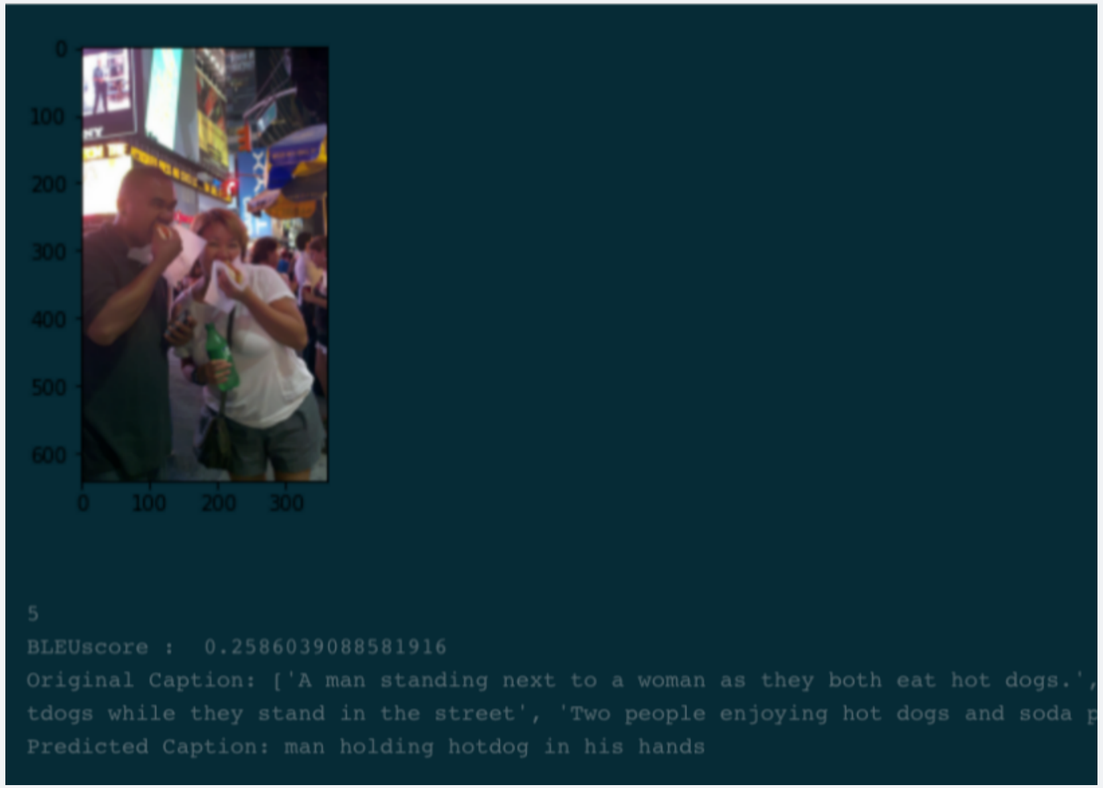
man holding hotdog in his hands

한국어 캡션

남자가 샌드위치를 입 베어 먹고

RESULTS

영어 캡션



man holding hotdog in his hands

한국어 캡션

무리의 사람들이 닌텐도 위 컨트롤러와 서

RESULTS

영어 캡션

group of people riding skis
on top of snow covered slope

한국어 캡션

스노우 보드를 타는 사람이 눈 속에

망한 결과 경진대회

1위 : 공포형

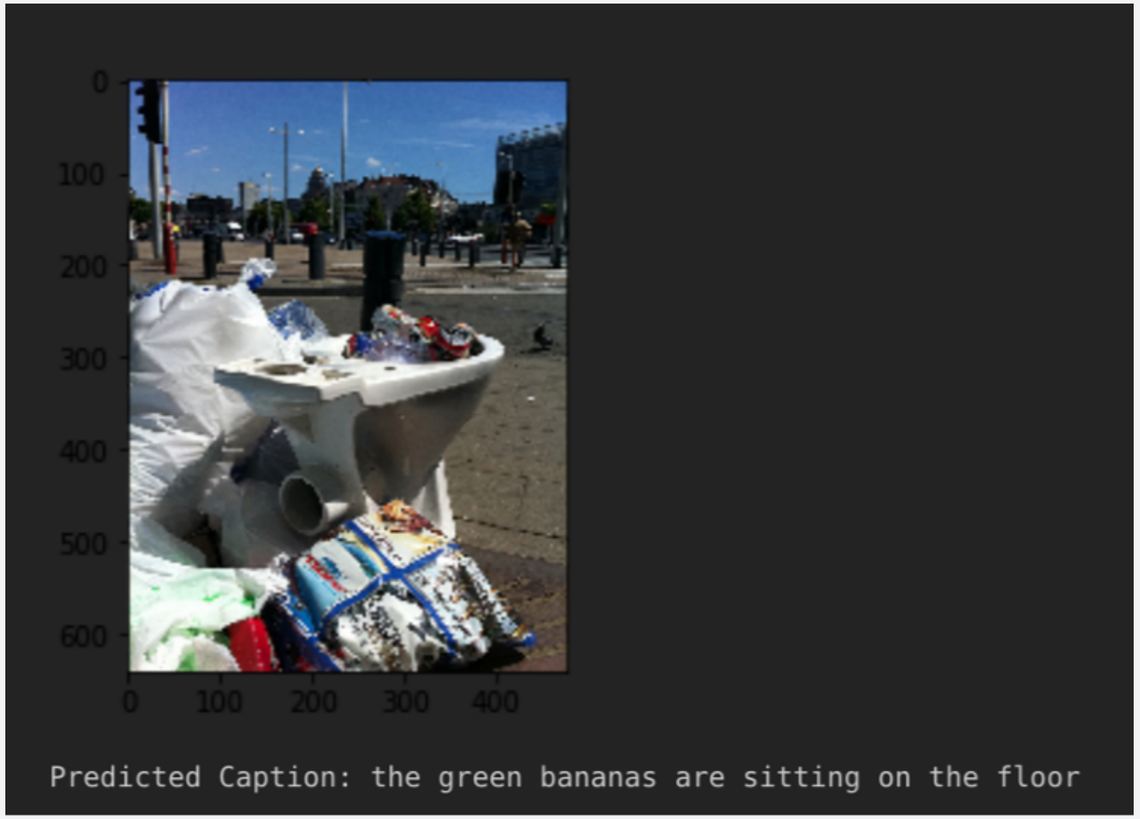
man is sitting on bench with his head

2위 : 반복형

소녀가 소녀가 위에 있는 있다 있다 있다 있다 있다….

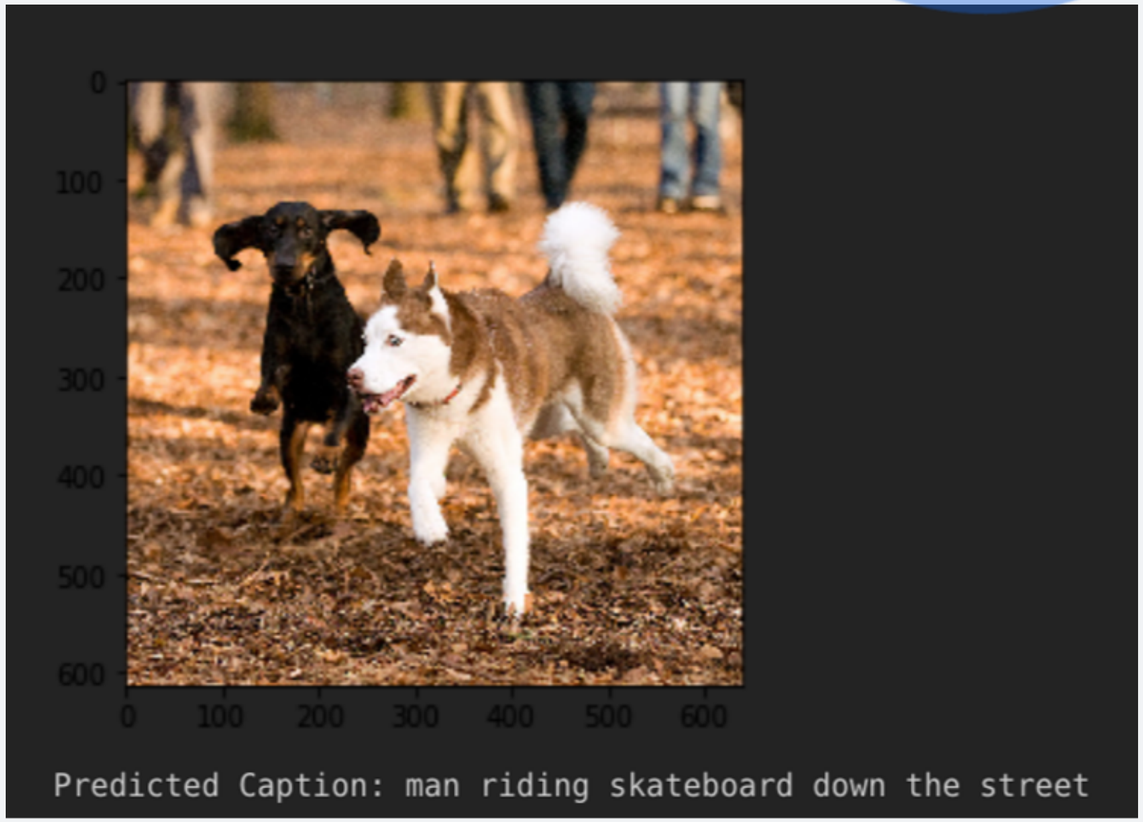
망한 결과 경진대회

3위 : 바보형



The green bananas are sitting on the floor

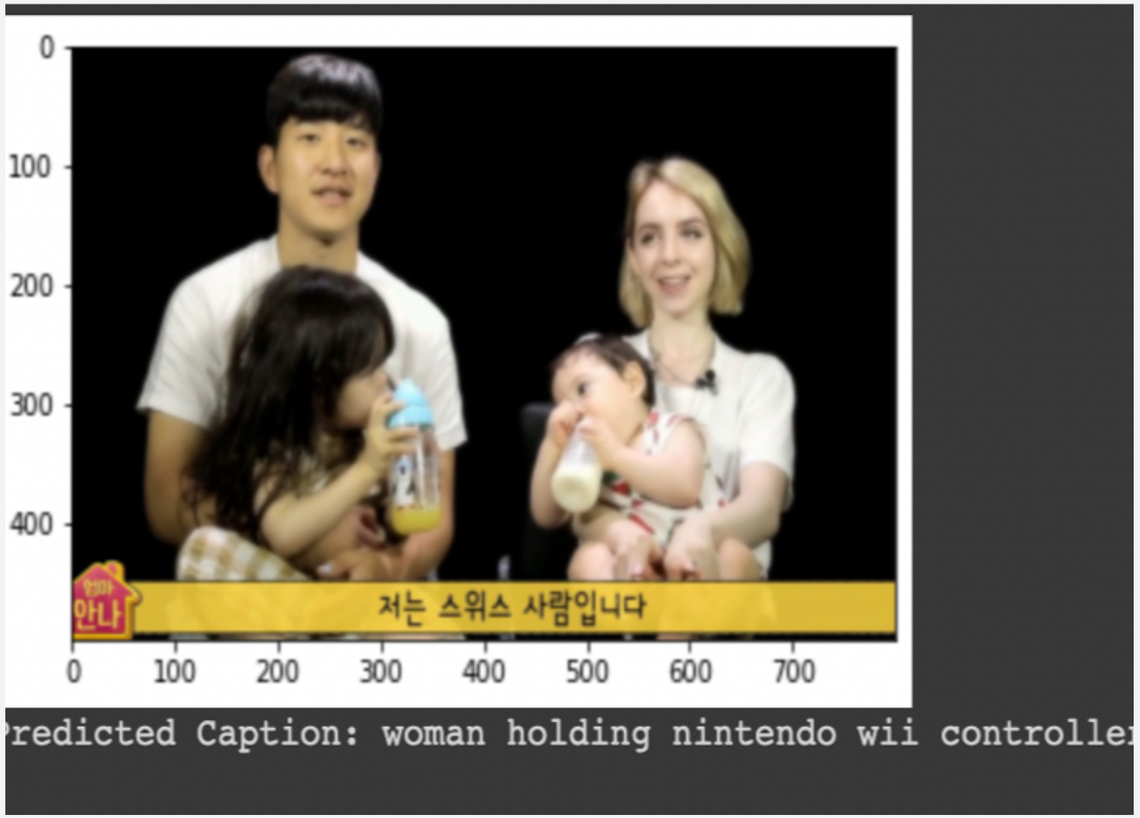
4위 : 관계 없음 형



Man riding skateboard down the street

망한 결과 경진대회

5위 : 어떻게 건후를 형



Woman holding Nintendo Wii controller

6위 : 머리 넘기잖아 형

여자가 휴대 전화를 이야기하며

03

CAPSULER

서비스 소개

이미지를 언어라는 캡슐에 담다.

WEB PAGE

WEB SKILLSET

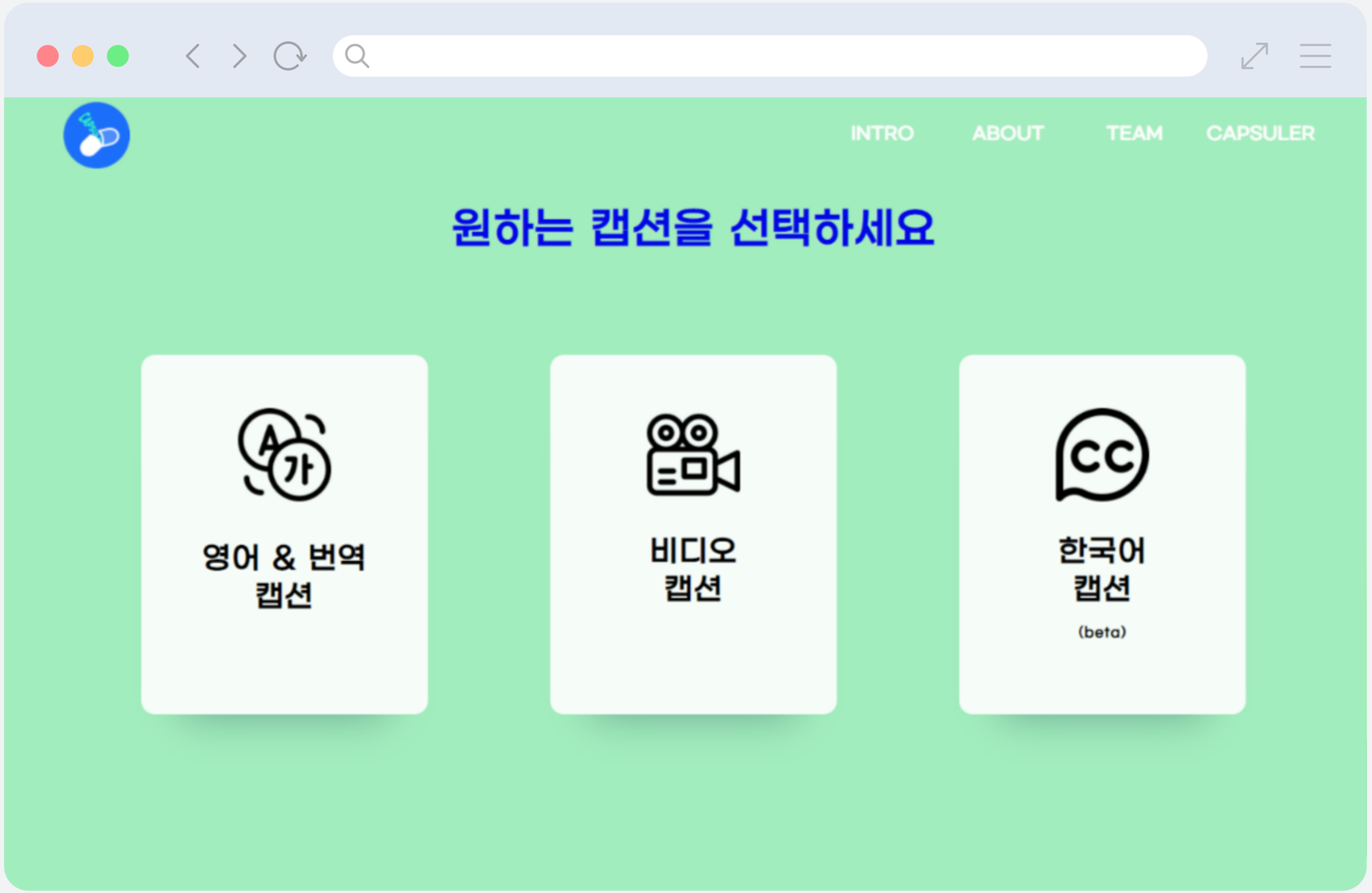
01 **Web Framework**
→ django

02 **Front-End**
→ html, CSS3, jQuery, javascript

03 **Back-End**
→ python

04 **Icons & Templates**
→ Flaticon

WEB PAGE

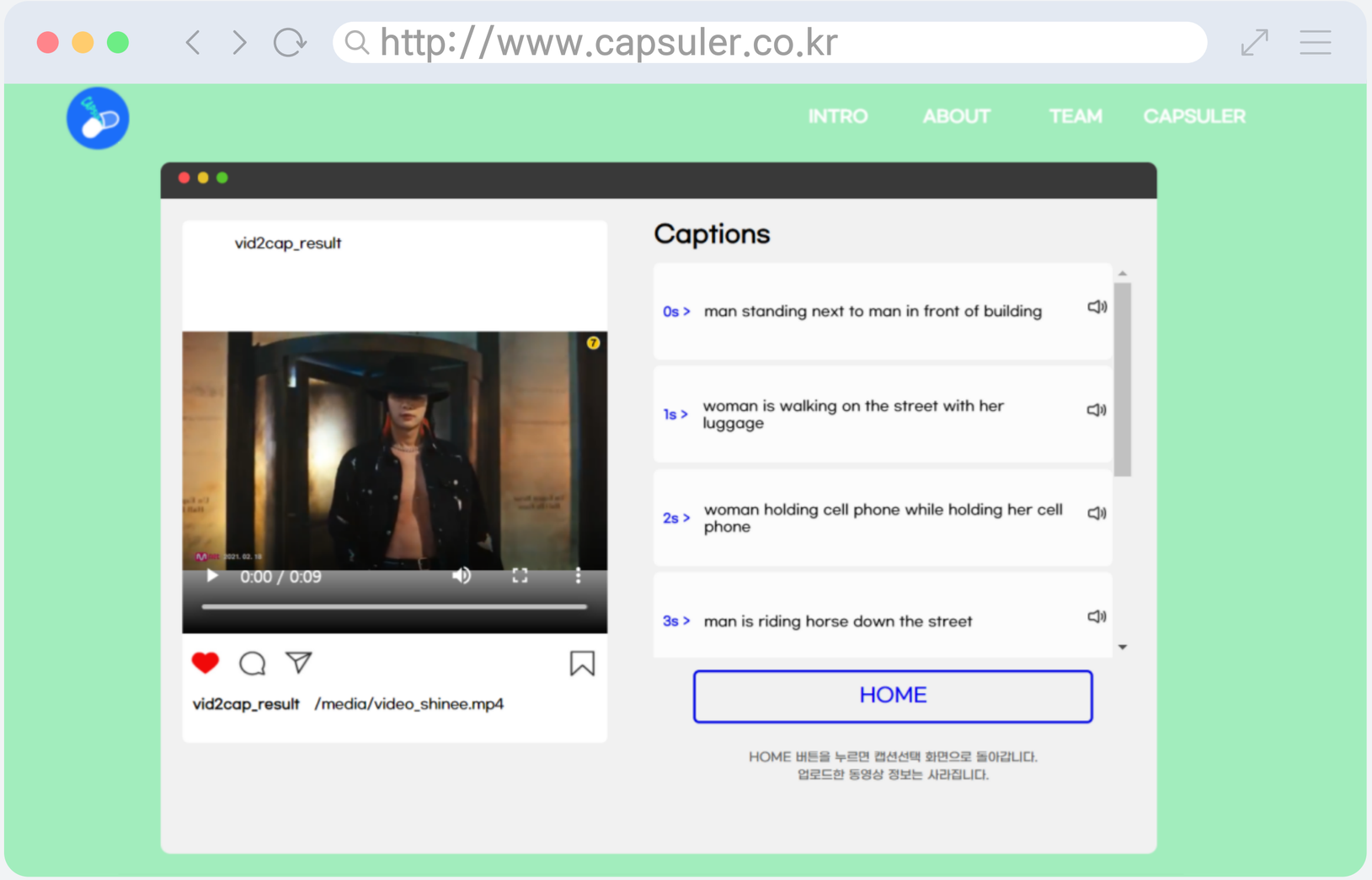


01 이미지 캡셔닝
02 영상 캡셔닝
03 해시태그

WEB PAGE

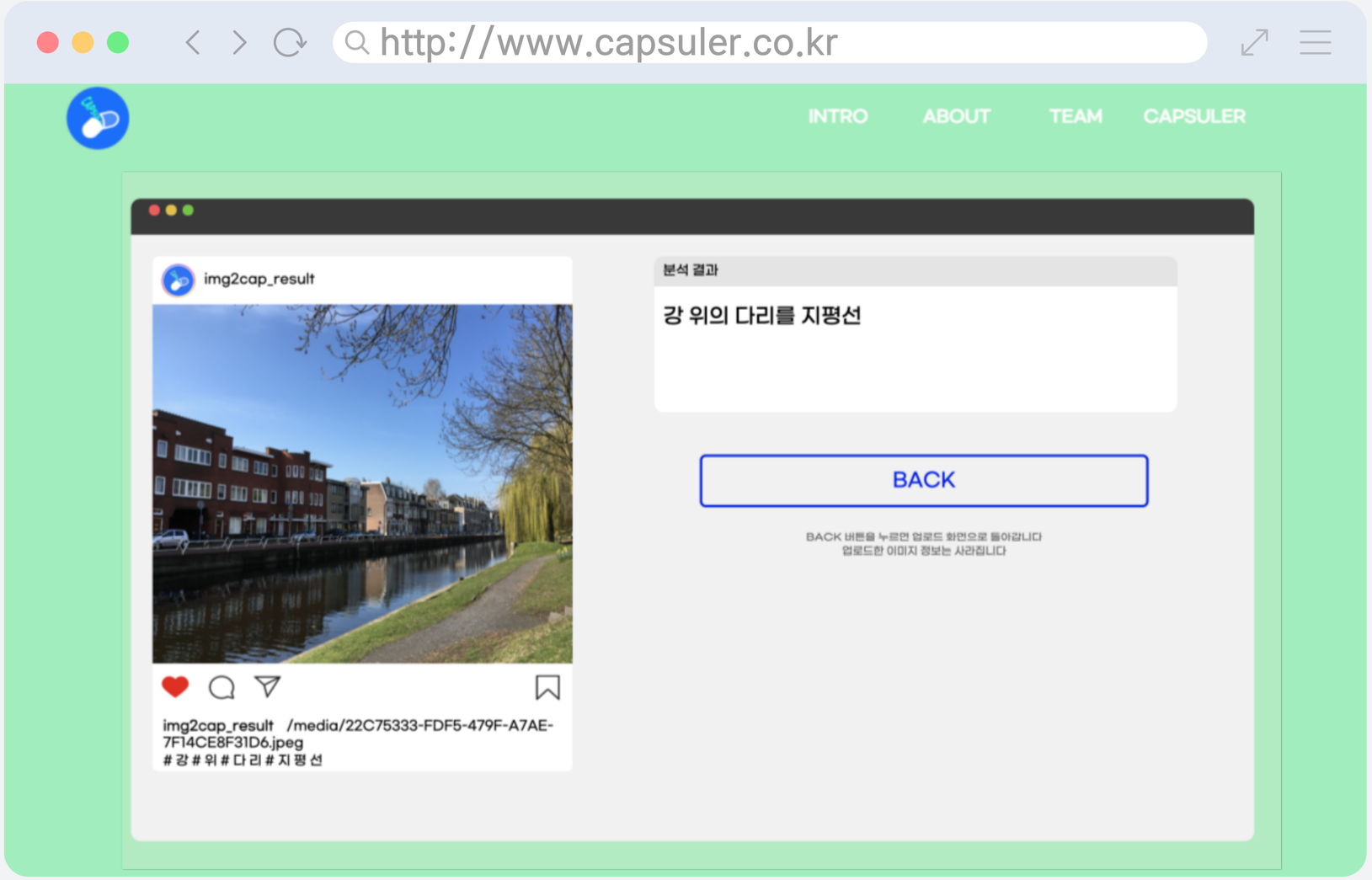
한/일/중/프
이미지 캡셔닝

WEB PAGE



영상 캡셔닝

WEB PAGE



한국어 해시태그

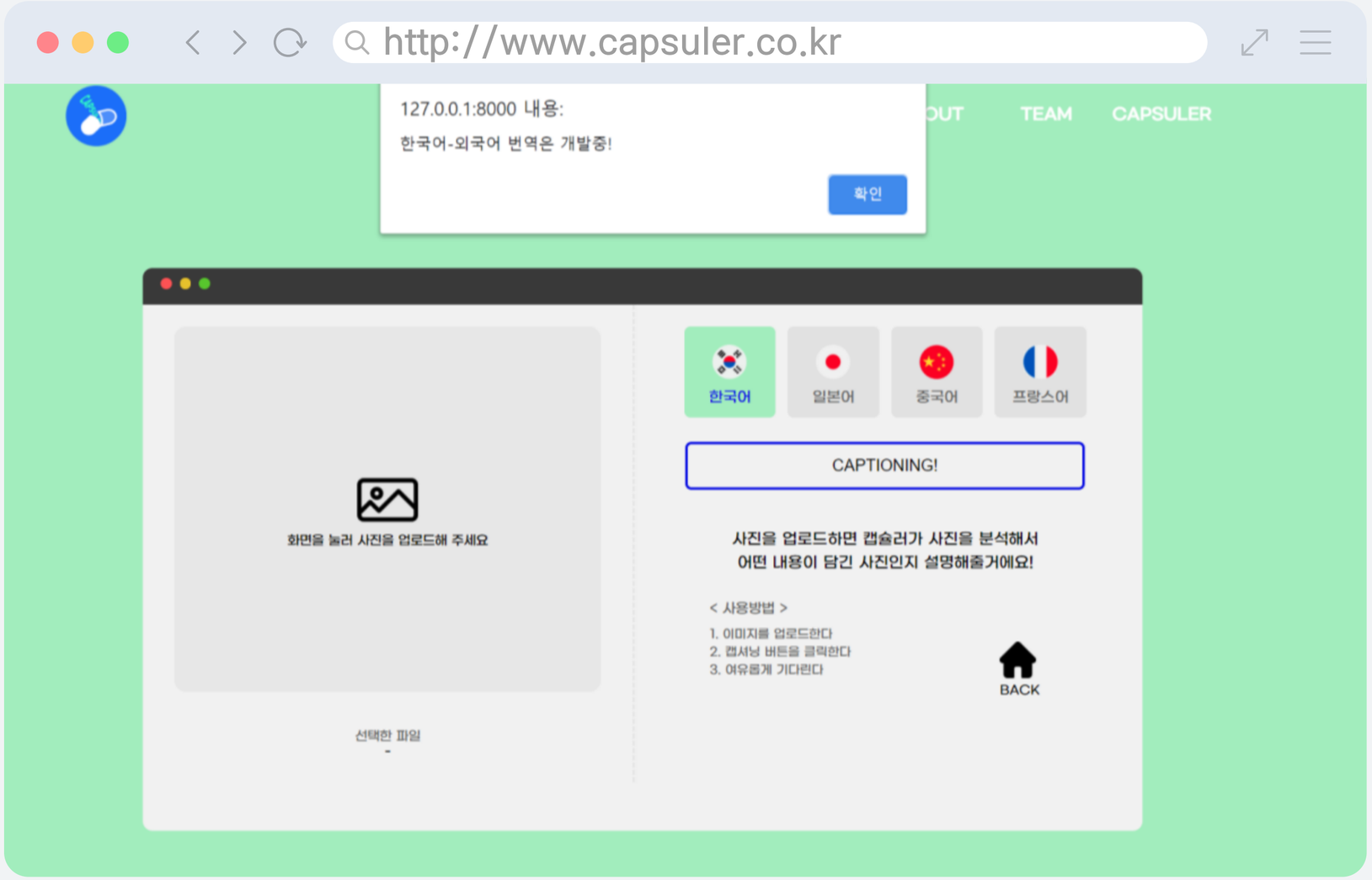
WEB PAGE

번역 & TTS

WEB PAGE

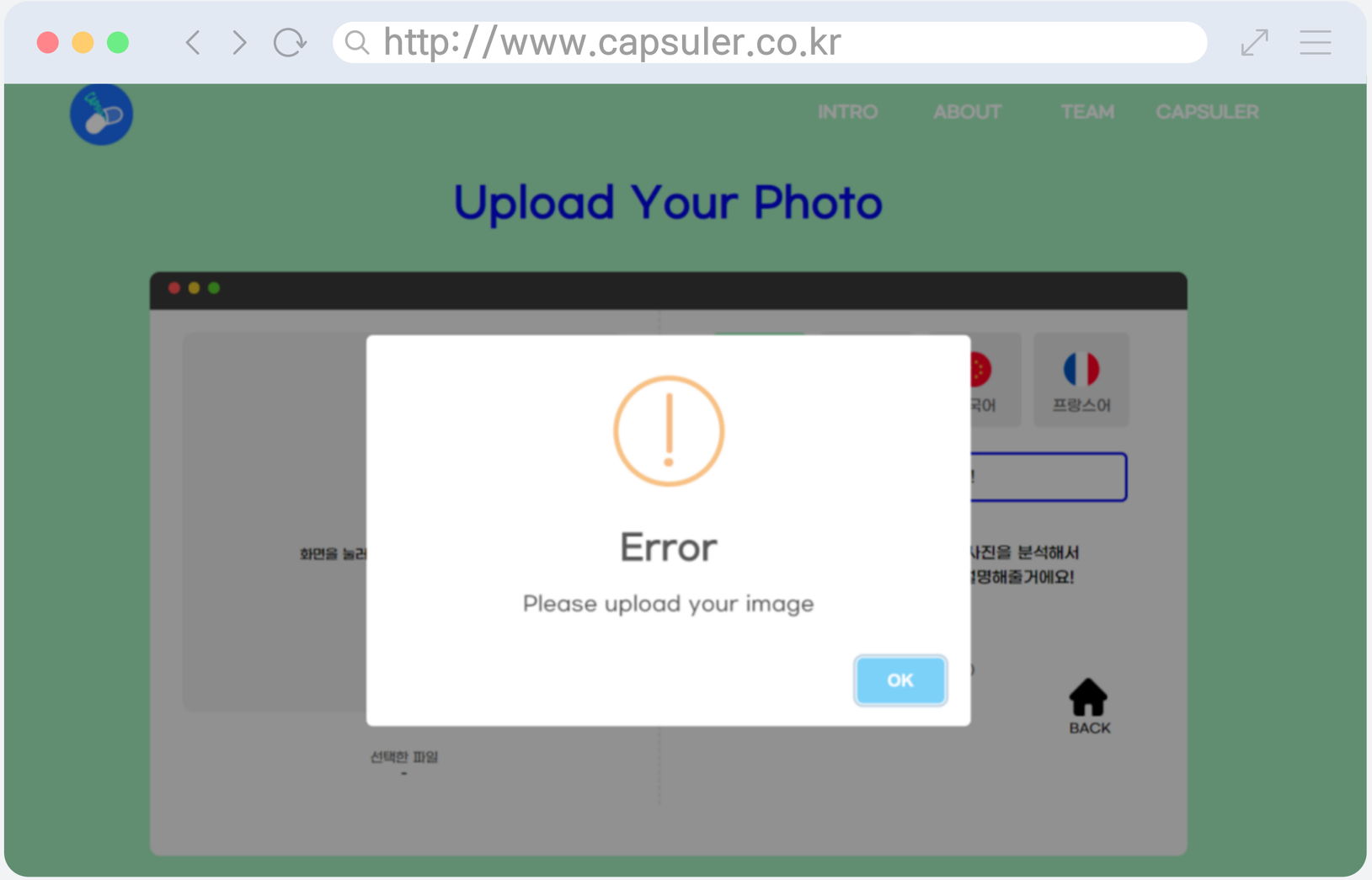
LOADING PAGE

WEB PAGE



ERROR MESSAGE

WEB PAGE



ERROR MESSAGE

04

CAPSULER

향후과제 & 소감

이미지를 언어라는 캡슐에 담다.

향후 활용 방안

01 문장 기반 영상 검색

다량의 영상/동영상 클립에서 원하는 영상 또는 영상
의 일부를 문장을 통해 검색

02 이미지 태깅

대량의 이미지 데이터셋 자동 생성
데일리 로그 자동 생성

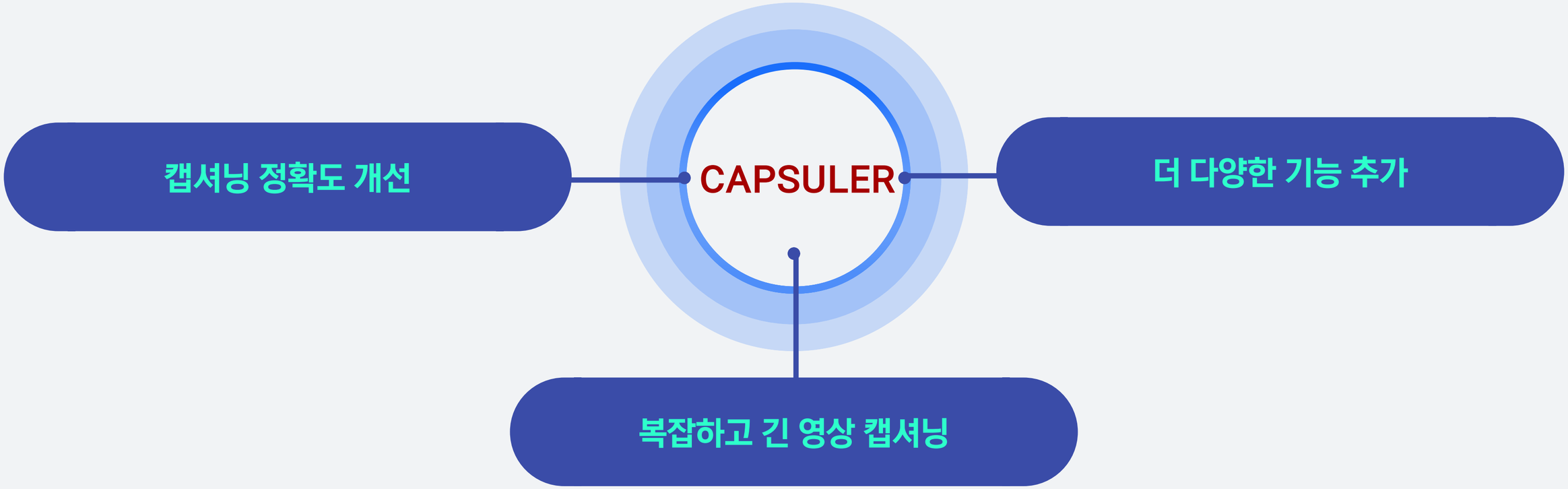
03 비전 정보기반 상황 설명

시각 장애인 보조 서비스
자동차 주행 중 측/후면 상황 설명

04 영상 텍스트화

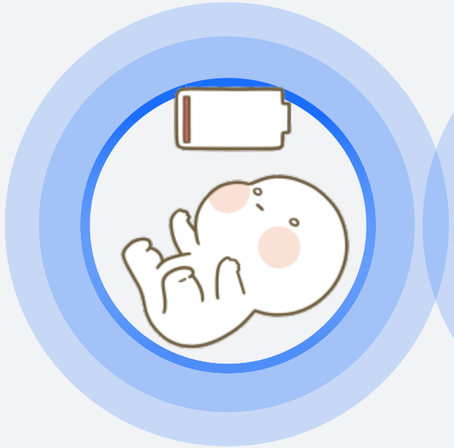
가정용/보안용 CCTV영상 텍스트화(음성화)
영상 컨텐츠 자막 자동 생성
영상 컨텐츠 줄거리 요약

향후 과제



소감

TEAM CAPSULER



강하은

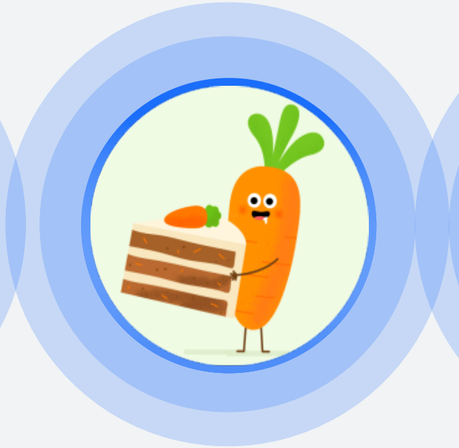
좋은 팀원들을 만나 즐겁게 일했습니다 ~~! 다음에도 같이 프로젝트 했으면 좋겠어요🔥

김수민

능력도 좋고 유머도 만점인 우리팀!!✨ 마지막 2주를 즐겁고 보람차게 마무리할 수 있었습니다 감사합니다!! 곧 피자먹으러갑시당🍕

김수희

💕앗 팀프로젝트 너무너무 재미있다  파이널에 걸맞는 즐거운 시간이었습니다 다들 만수무강💕



당현선

이렇게 재밌게 진행했던 팀프로젝트는 처음이었습니다.😎다같이 피자먹는 그날까지..🍀

이재준

팀원 케미 좋아서 정말 즐거운 파이널 프로젝트였습니다!!🙂 오프라인 정모도 할 수 있길 있다 있는 있다 있다 있는

CAPSULER

C A P S U L E R

THANK YOU

IMAGE READER, CAPSULER