MINIPROJECT
수어번역

수어영상 분석

팀명 - 12월 28일

박채은 이정룡 이상헌

CONTENTS

BITCAMP AI

TIME TABLE

사용데이터

DATA 1

일상 생활에서 자주 사용되는
선별된 단어 495개에 대한 수어동영상

자료출처
https://www.aihub.or.kr/mypage/reqst/datareqst/view.do?currMenu=157&topMenu=106&dataReqstSn=439783

DATA 2

일상 생활에서 자주 사용되는
선별된 문장 105개에 대한 수어동영상

자료출처
https://www.aihub.or.kr/mypage/reqst/datareqst/view.do?currMenu=157&topMenu=106&dataReqstSn=439786

사용 모델

MODEL1.

CRNN

고차원 데이터를 사용하기에
이상적인 CNN모델과
시간 시퀀스 데이터를
처리하는 GRU모델의 장점을
결합한 하이브리드 구조 사용.

초기 사용 모델

MODEL2.

MOVINETS

TENSORFLOW에서 만든
영상 분석을 위한 새로운 모델.
제한된 환경에서 대용량의
영상데이터를 빠르게
처리하기 위해 이용

최종 모델

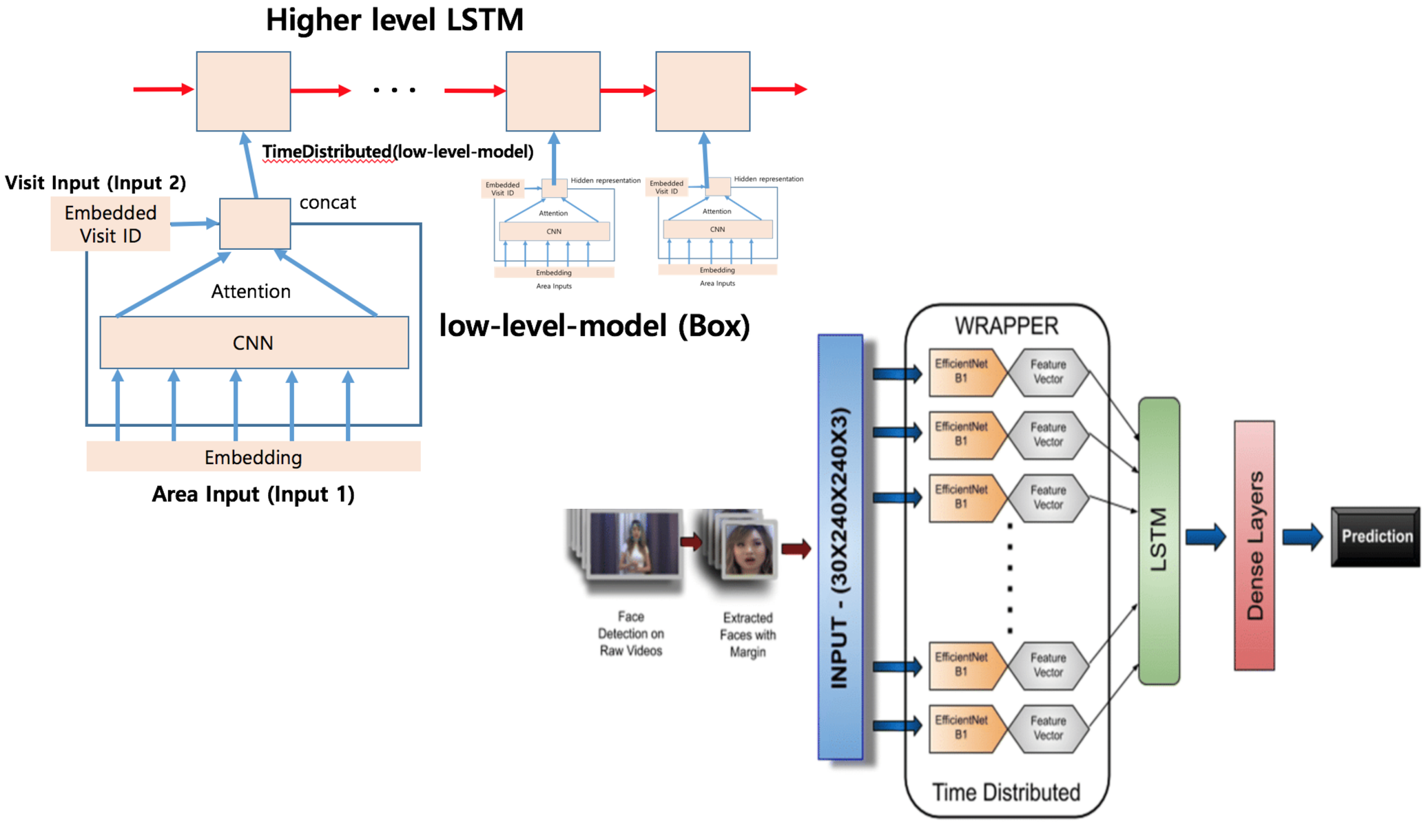
초기 사용 모델 (CRNN)

```python
def create_model():
    CNN_model = Sequential()
    CNN_model.add(Conv2D(16, (3, 3), input_shape=(70, 50, 3), activation='swish'))
    CNN_model.add(Conv2D(32, (3, 3), activation='swish'))
    CNN_model.add(Conv2D(16, (3, 3), activation='swish'))
    CNN_model.add(Conv2D(8, (3, 3), activation='swish'))
    CNN_model.add(Conv2D(8, (3, 3), activation='swish'))
    CNN_model.add(Conv2D(16, (3, 3), activation='swish'))
    CNN_model.add(Conv2D(16, (3, 3), activation='swish'))

    time_dist_CNN_model = Sequential()
    time_dist_CNN_model.add(TimeDistributed(CNN_model, input_shape=(sequence_len, 70, 50, 3)))

    model = Sequential()
    model.add(time_dist_CNN_model)              #  이대로 하면 LSTM에 입력되는 데이터가 (batch_size, timesteps, height, width, chanels) (5차원)
    model.add(TimeDistributed(Flatten()))       #  LSTM에 입력되는 데이터의 형태가 (batch_size, timesteps, features) (3차원)
    model.add(GRU(64,))
    model.add(Dense(32, activation='swish'))
    model.add(Dense(64, activation='swish'))
    model.add(Dense(length, activation='softmax'))
    model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model
```

TIME DISTRIBUTED



초기 사용 모델 (CRNN)

```
model1 = create_model()
model1.fit(X1, y1, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
model2 = create_model()
model2.fit(X2, y2, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
model3 = create_model()
model3.fit(X3, y3, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
model4 = create_model()
model4.fit(X4, y4, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
model5 = create_model()
model5.fit(X5, y5, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
model6 = create_model()
model6.fit(X6, y6, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
model7 = create_model()
model7.fit(X7, y7, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
model8 = create_model()
model8.fit(X8, y8, epochs=30, batch_size=5, shuffle=True, callbacks=[es], validation_split=0.2)
K.clear_session()
```

```
y_pred = ensemble([model1, model2, model3, model4, model5, model6, model7, model8], X_test)
y_pred = np.argmax(y_pred, axis=1)
```

초기 모델 사용 결과 (CRNN)

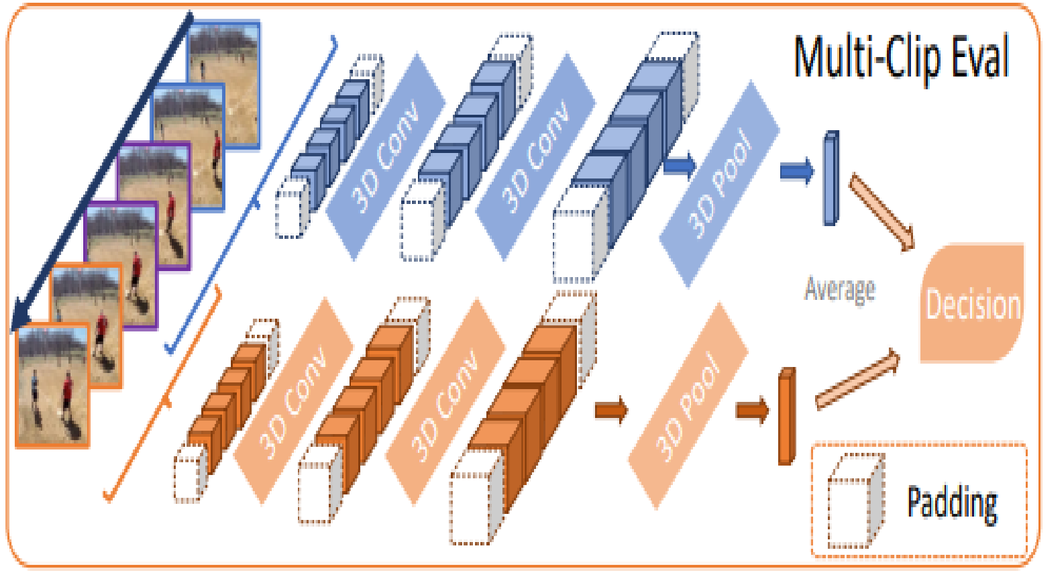
```
Epoch 1/1000
2024-02-27 20:26:00.364260: W tensorflow/core/kernels/gpu_ut
ils.cc:50] Failed to allocate memory for convolution redzone
 checking; skipping this check. This is benign and only mean
s that we won't check cudnn for out-of-bounds reads and writ
es. This message will only be printed once.
2024-02-27 20:26:01.828910: F tensorflow/stream_executor/cud
a/cuda_dnn.cc:593] Check failed: cudnnSetTensorNdDescriptor(
handle_.get(), elem_type, nd, dims.data(), strides.data()) =
= CUDNN_STATUS_SUCCESS (9 vs. 0)batch_descriptor: {count: 37
500 feature_map_count: 64 spatial: 32 32  value_min: 0.00000
0 value_max: 0.000000 layout: BatchDepthYX}
PS C:\Study>
```

```
loss 8.715789794921875
acc 0.014599974012374
시간 :  2714.4815421104431
```

인풋데이터 (x 데이터)를 8개로 나누고 배치사이즈를 아주 작게 줄였음에도 메모리 문제 발생.

정확도 매우 낮음

MOVINETS 모델 (BASE)

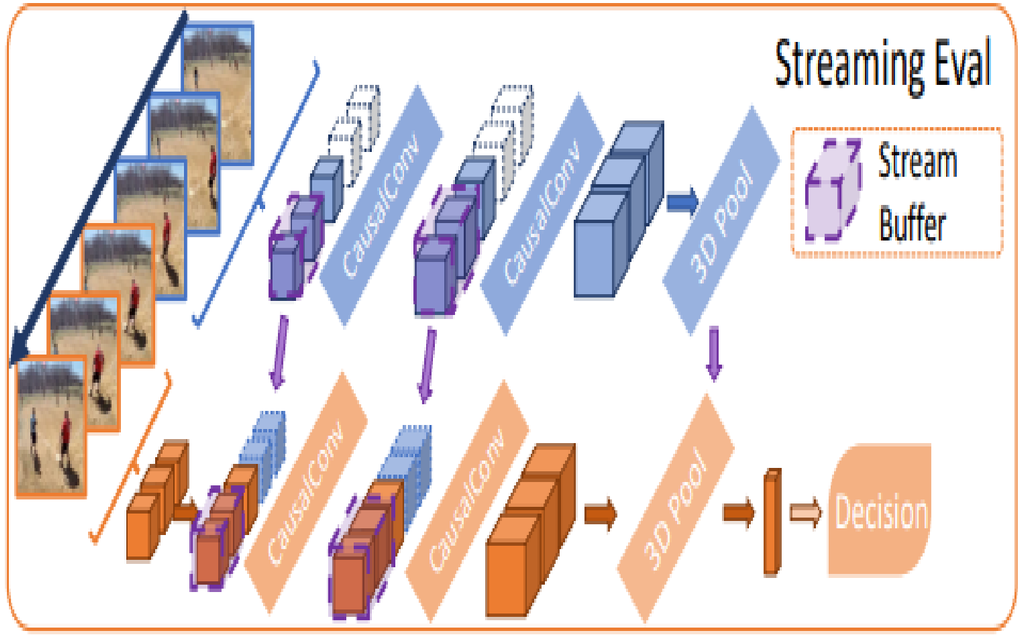


- 각각의 서브 클립마다
3D Conv를 통해 예측

- 예측값들을 평균내서
최종 예측값 산출

출처 : MoViNets: Mobile Video Networks for Efficient Video Recognition
arXiv:2103.11511

MOVINETS 모델 (STREAM)



- 스트림 버퍼 사용으로 자원을 효율적으로 사용.

- 스트림 버퍼 사용시 메모리 사용량이 약 40프로 감소

- 데이터 로딩 지연시간 감소.

출처 : MoViNets: Mobile Video Networks for Efficient Video Recognition
arXiv:2103.11511

MOVINETS 모델

CONFIG	FULL INPUT	BATCH SIZE	Memory	TOP-1
No Buffer	16×172^2	8	5.8 GB	55.9
Buffer (8 frames)	16×172^2	16	6.6 GB	58.5
No Buffer	16×172^2	16	10.4 GB	OOM

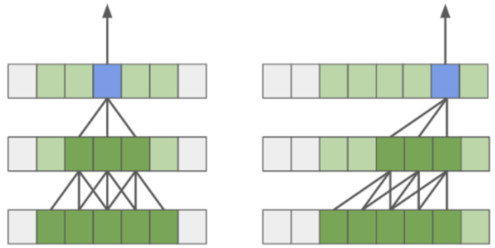


Figure 7. **Standard Convolution vs. Causal Convolution**. The figure illustrates the effective receptive field along a sequence of frames. The temporal kernel size is 3, with padding shown in white. Causal convolution can be performed efficiently by padding only on one side of the time axis thus to force the output causality.

- Causal Convolution 사용시 한쪽 방향으로만 패딩 적용.

- 과거 데이터, 현재 데이터만 이용하여 현재의 출력 계산. 미래의 데이터 없이 현재의 출력 계산.

- 순차적 데이터를 처리하는데 용이.

출처 : MoViNets: Mobile Video Networks for Efficient Video Recognition
arXiv:2103.11511

MOVINETS 모델 핵심코드

```
class FrameGenerator:
  def __init__(self, path, n_frames, training = False):
    self.path = path
    self.n_frames = n_frames
    self.training = training
    self.class_names = sorted(set(p.name for p in self.path.iterdir() if p.is_dir()))
    self.class_ids_for_name = dict((name, idx) for idx, name in enumerate(self.class_names))

  def get_files_and_class_names(self):
    video_paths = list(self.path.glob('*/*.mp4'))
    classes = [p.parent.name for p in video_paths]
    return video_paths, classes

  def __call__(self):
    video_paths, classes = self.get_files_and_class_names()
    pairs = list(zip(video_paths, classes))

    if self.training:
      random.shuffle(pairs)

    for path, name in pairs:
      video_frames = frames_from_video_file(path, self.n_frames)
      label = self.class_ids_for_name[name] # Encode labels
      yield video_frames, label
```

MOVINETS 모델 핵심코드

```python
def frames_from_video_file(video_path, n_frames, output_size = (172,172), frame_step = 2):
  # Read each video frame by frame
  result = []
  src = cv2.VideoCapture(str(video_path))
  video_length = src.get(cv2.CAP_PROP_FRAME_COUNT)
  need_length = 1 + (n_frames - 1) * frame_step
  if need_length > video_length:
    start = 0
  else:
    max_start = video_length - need_length
    start = random.randint(0, max_start + 1)

  src.set(cv2.CAP_PROP_POS_FRAMES, start)
  # ret is a boolean indicating whether read was successful, frame is the image itself
  ret, frame = src.read()
  result.append(format_frames(frame, output_size))

  for _ in range(n_frames - 1):
    for _ in range(frame_step):
      ret, frame = src.read()
    if ret:
      frame = format_frames(frame, output_size)
      result.append(frame)
    else:
      result.append(np.zeros_like(result[0]))
  src.release()
  result = np.array(result)[..., [2, 1, 0]]
  return result
```

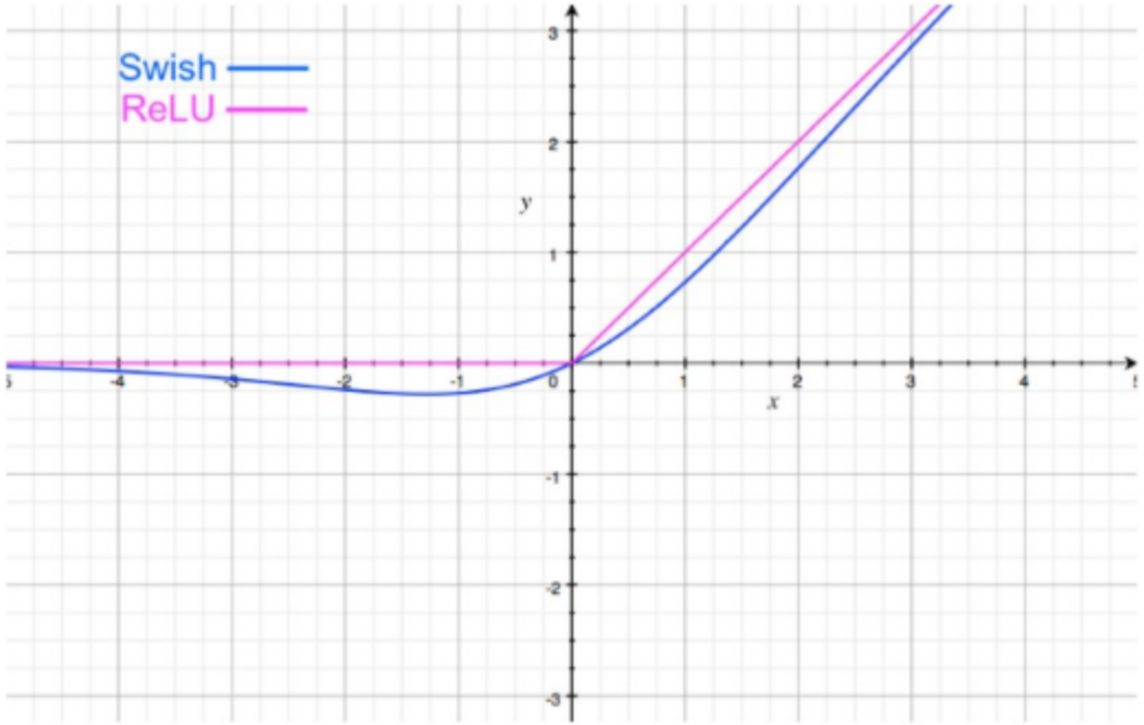
MOVINETS 모델 핵심코드

```
model_id = 'a0'
resolution = 172

backbone = movinet.Movinet(
    model_id=model_id,
    causal=True,
    conv_type='2plus1d',
    se_type='2plus3d',
    activation='hard_swish',
    gating_activation='hard_sigmoid',
    use_external_states=False,
)

model = movinet_model.MovinetClassifier(
    backbone,
    num_classes=600,
    output_states=True)
```

swish 활성화 함수



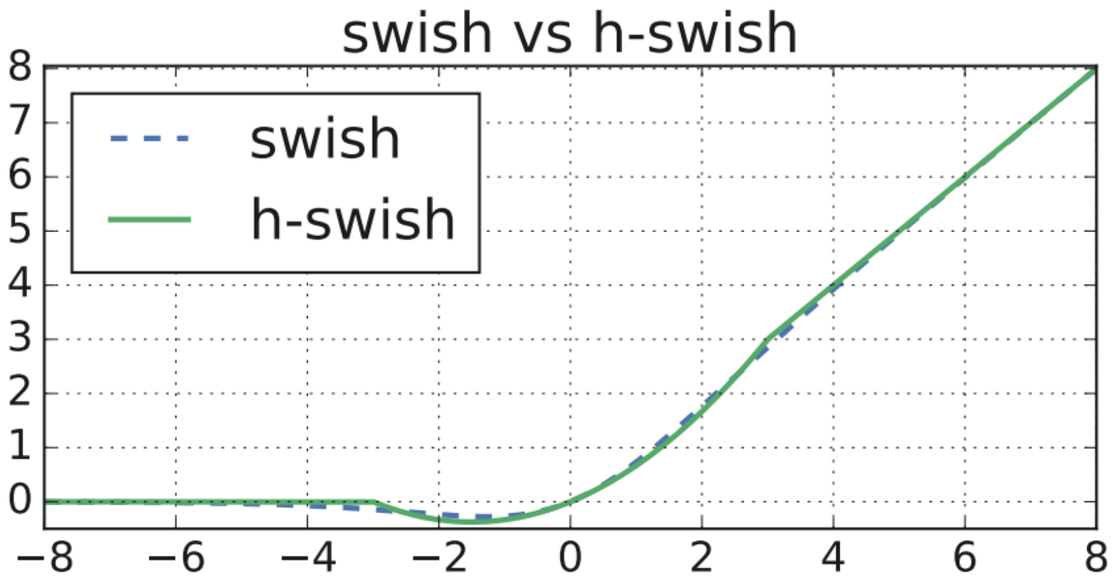
- relu를 대체하기 위하여 만든 함수.

- sigmoid 함수에 x를 곱한 형태.

- relu에 비해 깊은 레이어에서 효과적으로 작용.

출처 : https://kejdev.tistory.com/19

hard_swish 활성화 함수



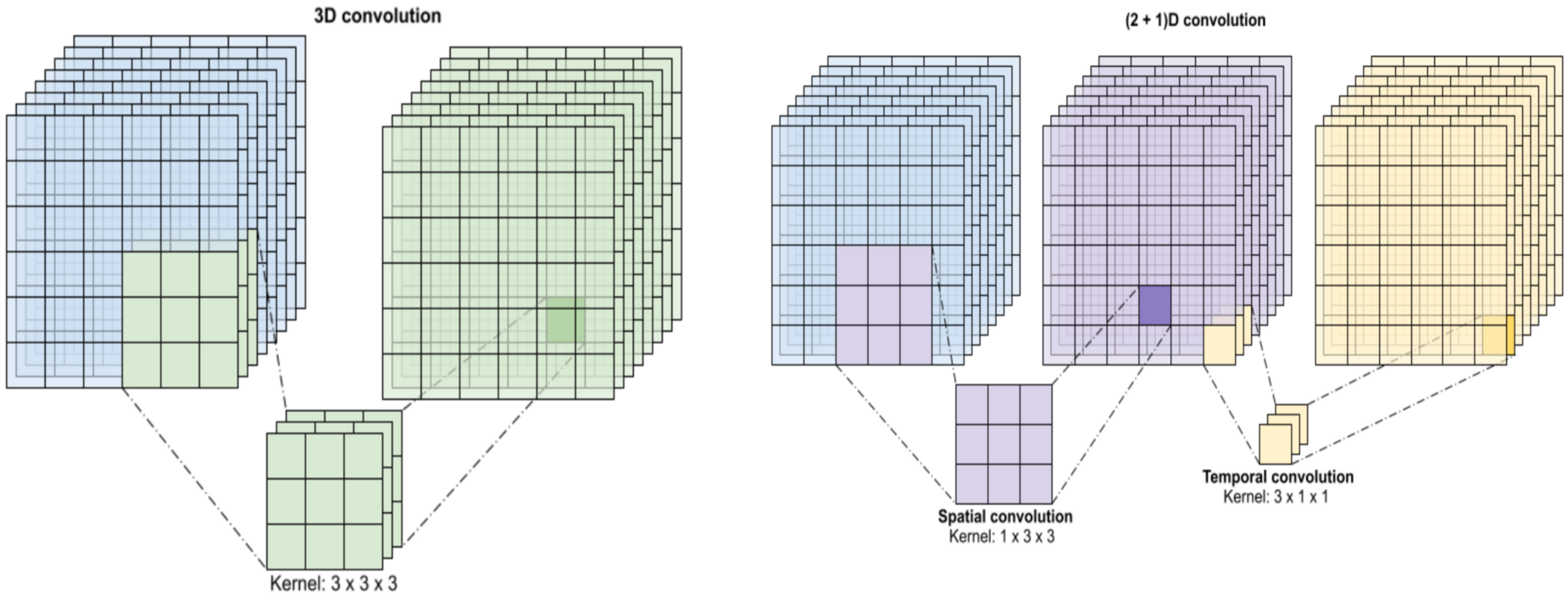
- 모델 경량화에 주로 이용

- swish에 비해 계산이 간단해
계산비용이 비교적 낮음

- 계산비용이 낮아
메모리 사용량 감소에 효과적

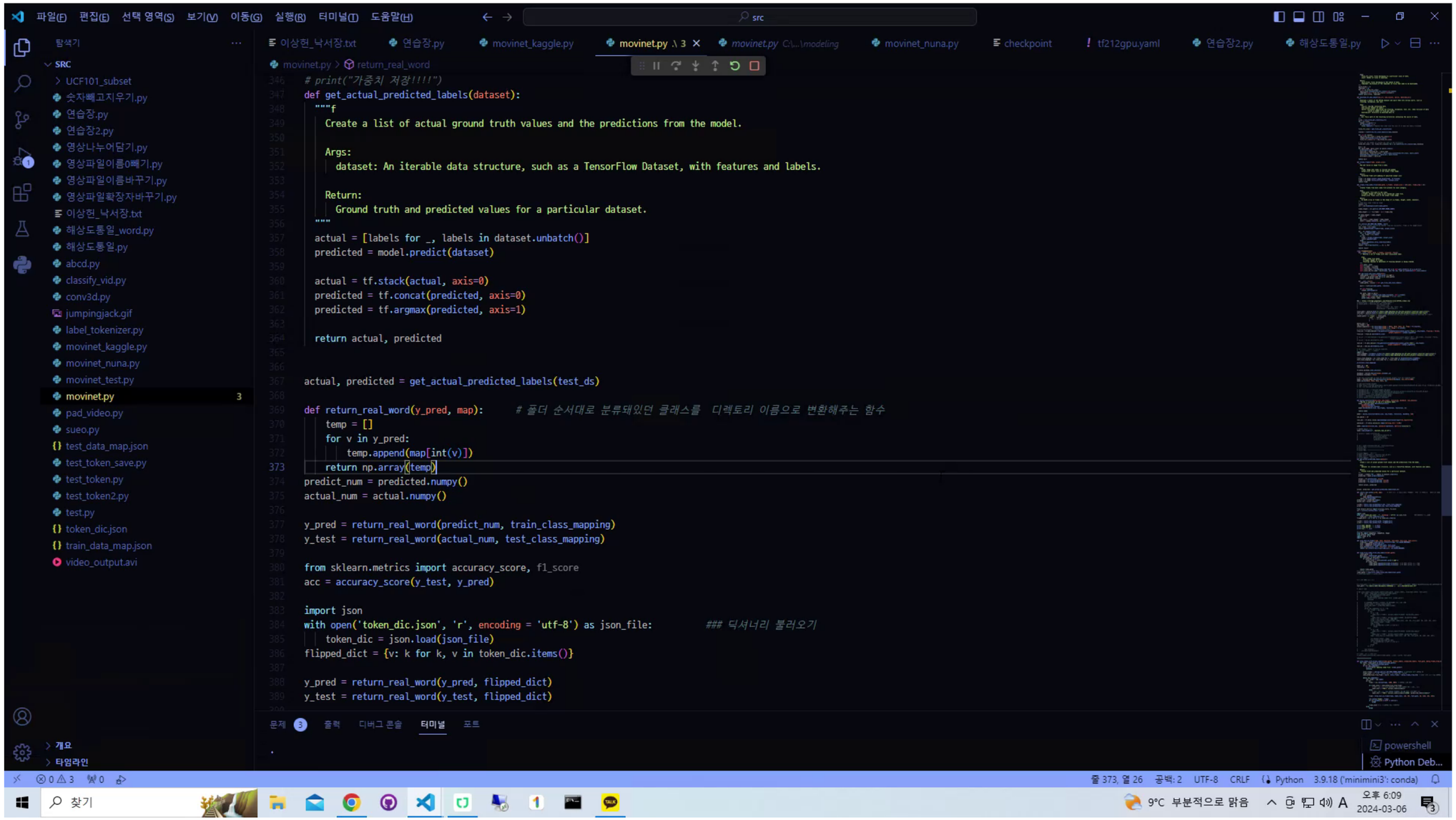
출처 : Searching for MobileNetV3

MOVINETS 모델 부연설명



출처 : https://www.tensorflow.org/tutorials/video/video_classification

MOVINETS 모델 최종결과



Q & A TIME

THANK YOU

박채은 이정룡 이상헌