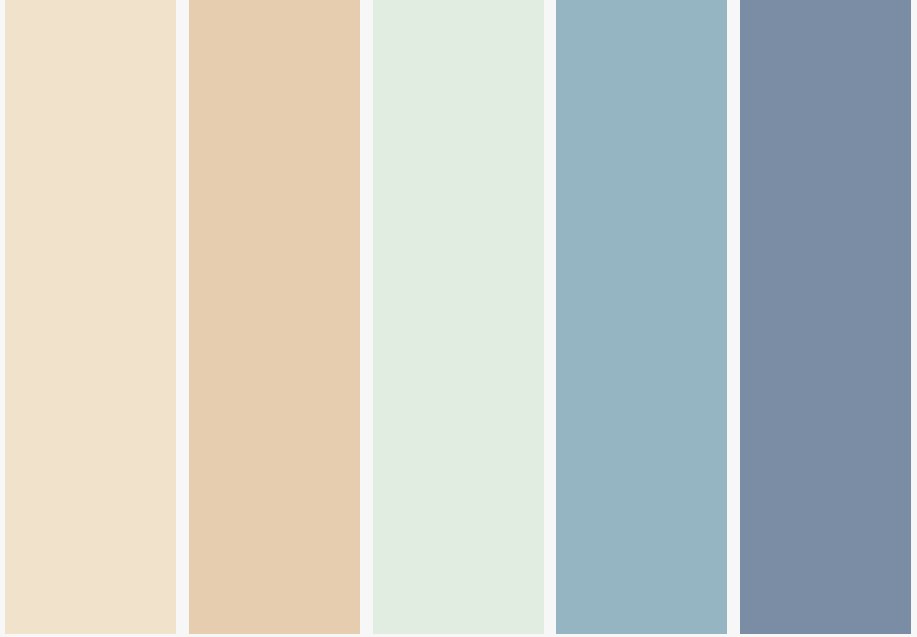


머신러닝 눈 깜빡임 인식을 통한 에너지 절약 장치 구현

[그린에너지특성화 관련 에너지 절약 IoT(사물인터넷) 제어 기술]

201824089 김하얀
201823001 맹하늘
201822181 장하영
201824413 김현정
201720769 신선영
201822437 최유진

INDEX

INDEX

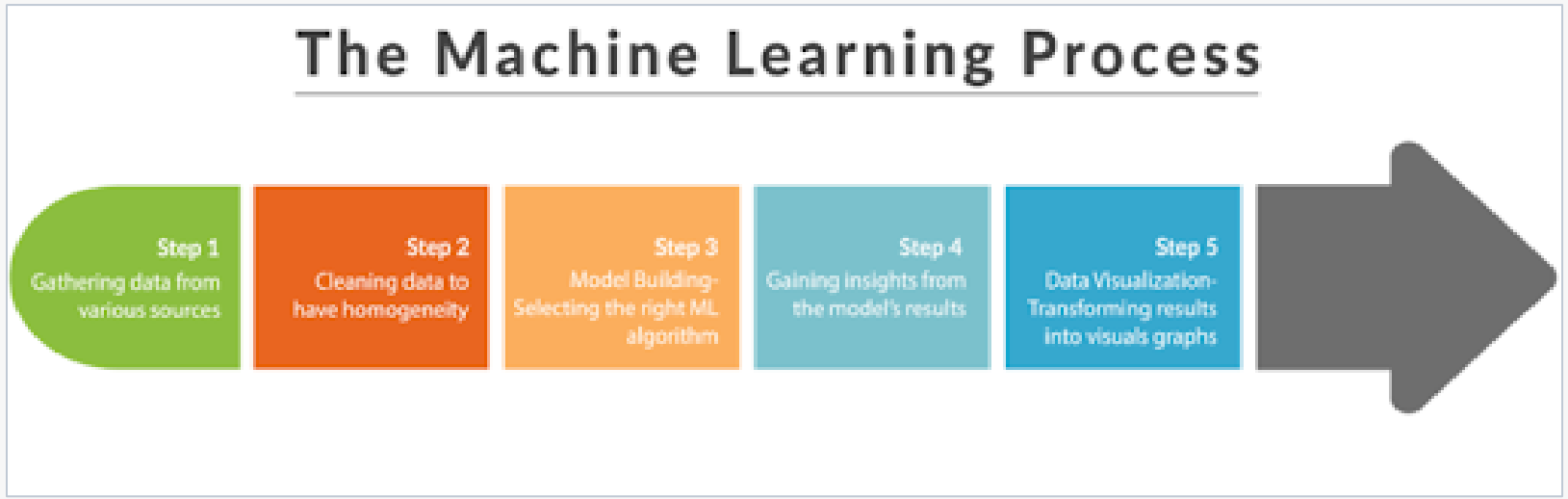
개 념

교과 학습 내용

머신러닝?

- 데이터를 이용하여 어떤 시직이나 패턴을 학습하는 것
- 경험적 데이터를 기반으로 학습, 예측 수행, 성능을 향상시키는 시스템과 이를 위한 알고리즘 연구하고 구축하는 기술

기능 수준	입력에 따른 출력의 변화 여부	레벨 3 : 머신러닝 (자연어 처리, 지도/비지도 학습)



[그림] 머신러닝 프로세스

개 념

교과 학습 내용

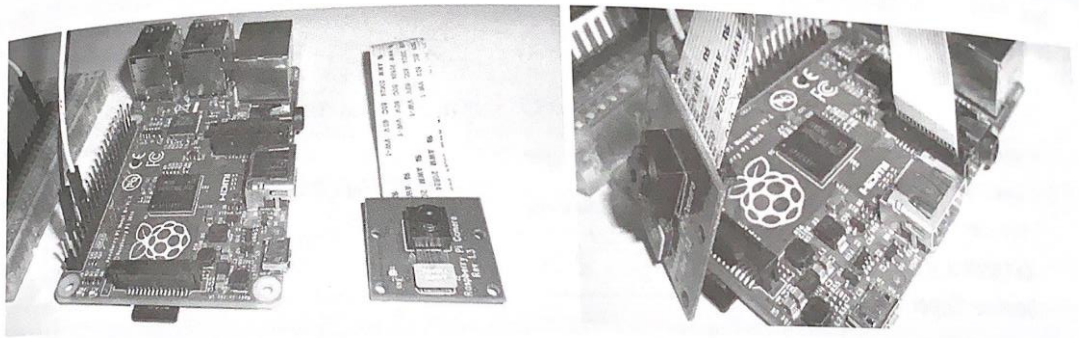


그림 7-37 Pi Camera의 설치

- LED 소자
  - 음극 – 라즈베리 파이 GND 핀
  - 양극 – 라즈베리 파이 GPIO
- 저항
  - 라즈베리 파이 GPIO 핀
  - LED 음극

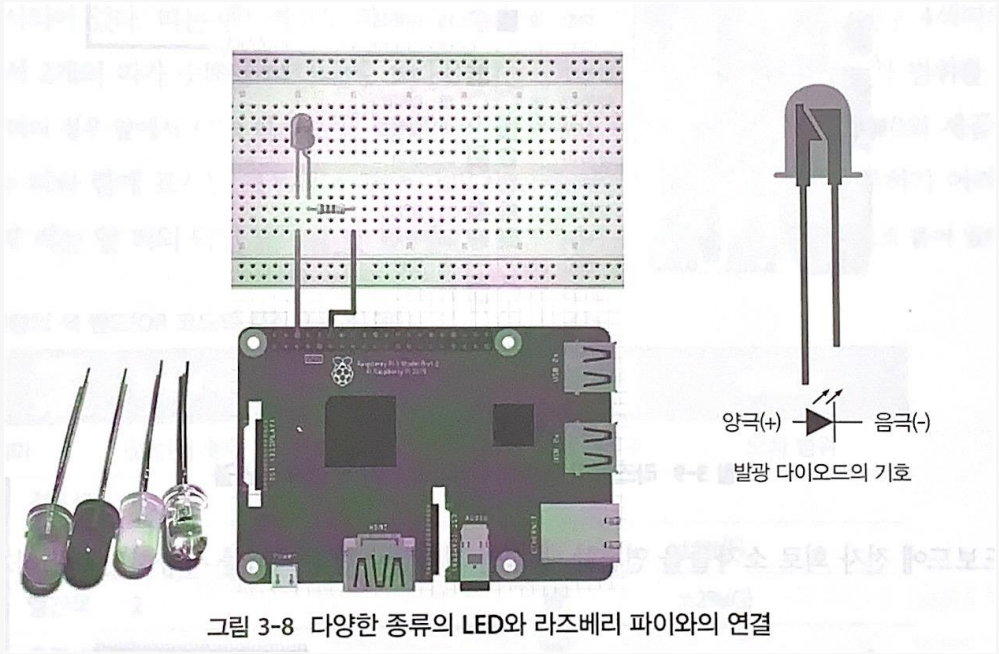


그림 3-8 다양한 종류의 LED와 라즈베리 파이와의 연결

- 라즈베리 전원 완전히 제거
- CSI 카메라 커넥터 고정핀 위로 올림
- Pi Camera 커넥터를 완전히 꽂고 다시 내림

개 념

작품 주요 내용

눈을 일정시간 이상 감고 있으면 자동으로 방의 불을 꺼준다.

선정 이유

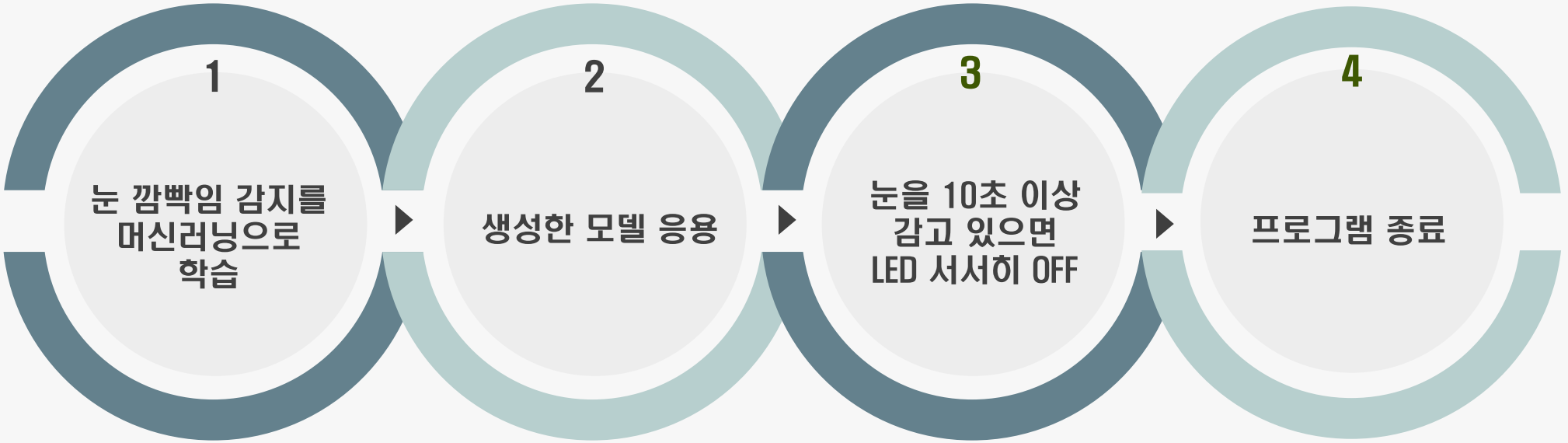
IoT, 머신 러닝,
에너지 절약 분야를
모두 접목하여
구현할 수 있는 시스템

기대 효과

에어컨, 선풍기, 보일러 등 가전
제품들을 켜고 잠드는 경우
타이머를 자동으로 시작하거나
가전제품을 자동으로 off 하여
낭비되는 전기 에너지절약

개 념

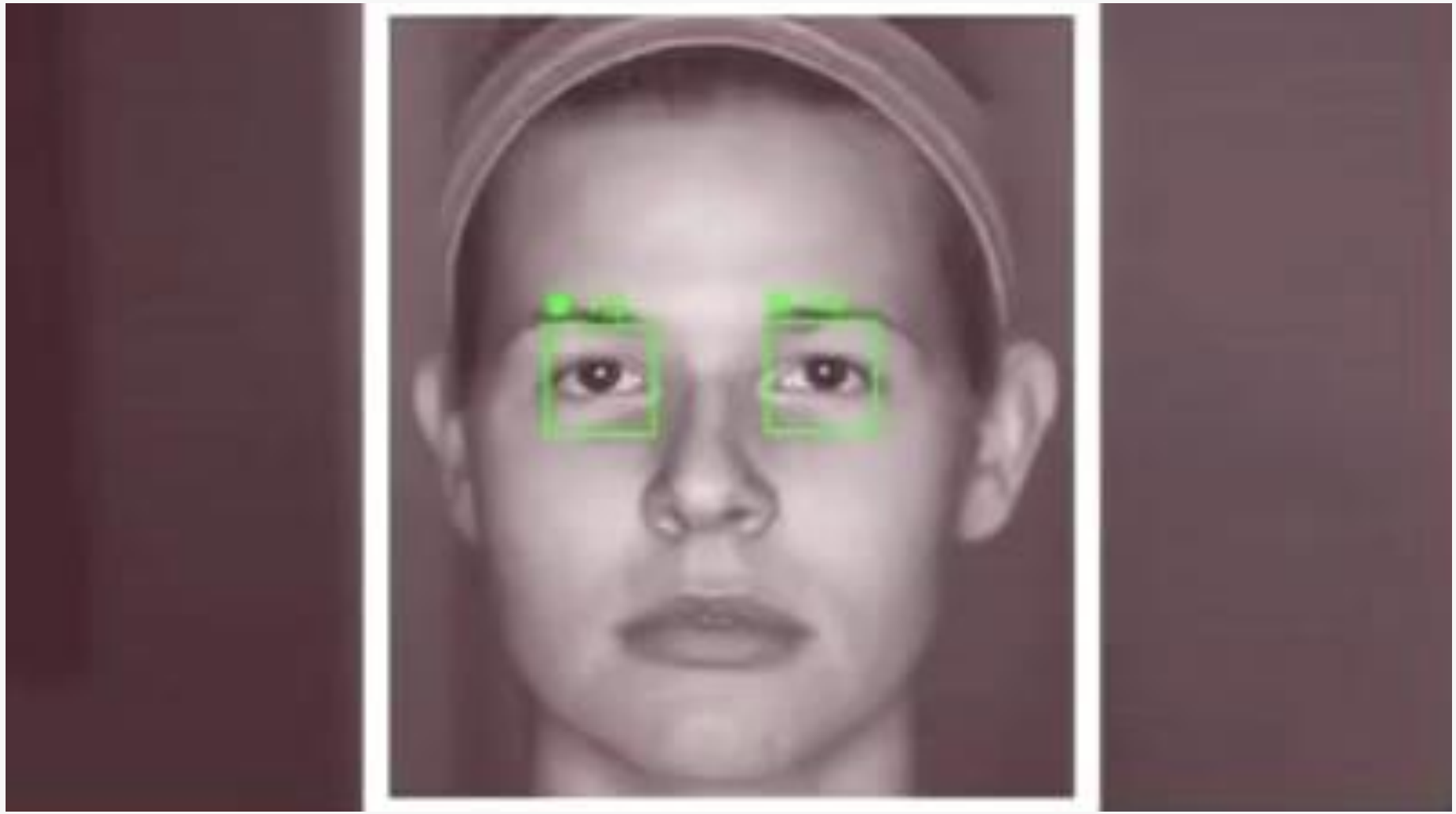
작품 주요 내용



⇒ 소스코드에 변수를 추가하여 눈을 감고 있는 기준 시간을 사용자가 직접 설정할 수 있다면 사용자 각각에게 맞는 설정 시간으로 시스템 사용이 가능

개 념

작품 주요 내용



INDEX

2 설계

설 계

작품 구성 요소

Pi Camera

라즈베리 파이 재단에서 제공하는 카메라
라즈베리 파이의 CSI 카메라 커넥터를 이용하여 사용 가능

TensorFlow

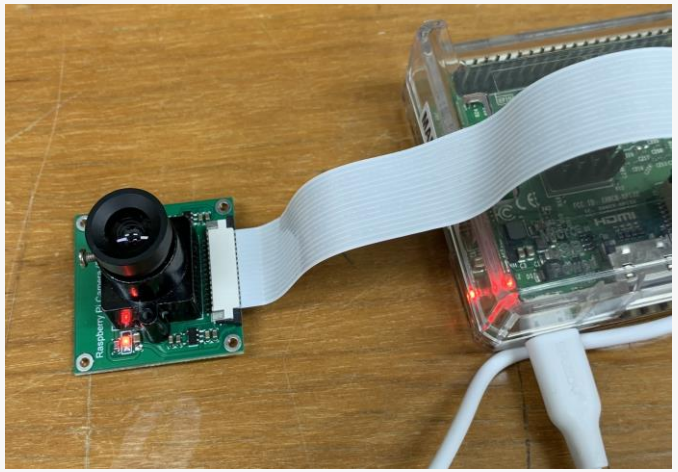
딥 러닝과 머신 러닝 등에 활용하기 위해 개발
된 오픈소스 소프트웨어

Keras

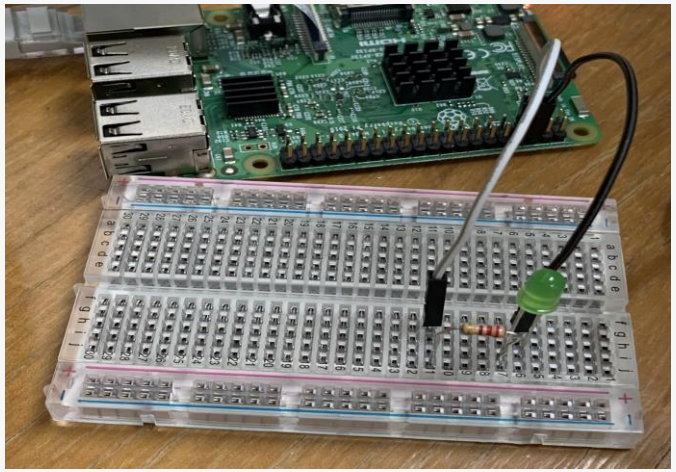
파이썬으로 작성된 오픈 소스 신경망 라이브러리
모듈의 독립적 설정 가능

OpenCV

쉬운 영상 처리가 가능한 라이브러리



Pi Camera



LED

설 계

주요 모듈

jupyter notebook	raspberry pi에서 jupyter notebook 실행 불가, windows 컴퓨터에서 jupyter notebook을 사용하여 dataset으로 머신러닝 학습
python 3.8 설치	sudo apt-get install python3
OpenCV2 설치	pip install opencv-python
Keras 설치	pip uninstall keras python -m pip install keras pip install keras==2.3.1
TensorFlow 설치	wget https://github.com/lhelontra/tensorflow-on-arm/releases/download/v2.0.0/tensorflow-2.0.0-cp37-none-linux_armv7l.whl python3 -m pip uninstall tensorflow python3 -m pip install tensorflow-2.0.0-cp37-none-linux_armv7l.whl
dlib 설치	CMake를 사용하여 빌드
matplotlib 설치	pip install matplotlib

설 계

눈 깜빡임 감지 머신러닝 소스코드

[1]
```
import datetime
import numpy as np
import matplotlib.pyplot as plt
from keras.layers import Input, Activation, Conv2D, Flatten, Dense, MaxPooling2D
from keras.models import Model, load_model
from keras.preprocessing.image import ImageDataGenerator
from keras.callbacks import ModelCheckpoint, ReduceLROnPlateau
plt.style.use('dark_background')
```

[2]
```
x_train = np.load('C:/Users/skt/Desktop/Twinkle/x_train.npy').astype(np.float32)
y_train = np.load('C:/Users/skt/Desktop/Twinkle/y_train.npy').astype(np.float32)
x_val = np.load('C:/Users/skt/Desktop/Twinkle/x_val.npy').astype(np.float32)
y_val = np.load('C:/Users/skt/Desktop/Twinkle/y_val.npy').astype(np.float32)

print(x_train.shape, y_train.shape)
print(x_val.shape, y_val.shape)
```

설 계

눈 깜빡임 감지 머신러닝 소스코드

[3]
```
plt.subplot(2, 1, 1)
plt.title(str(y_train[0]))
plt.imshow(x_train[0].reshape((26, 34)), cmap='gray')
plt.subplot(2, 1, 2)
plt.title(str(y_val[4]))
plt.imshow(x_val[4].reshape((26, 34)), cmap='gray')
```

[4]
```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=10,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2
)

val_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow(
    x=x_train, y=y_train,
    batch_size=32,
    shuffle=True
)

val_generator = val_datagen.flow(
    x=x_val, y=y_val,
    batch_size=32,
    shuffle=False
)
```

설 계

눈 깜빡임 감지 머신러닝 소스코드

[5]

```
inputs = Input(shape=(26, 34, 1))

net = Conv2D(32, kernel_size=3, strides=1, padding='same', activation='relu')(inputs)
net = MaxPooling2D(pool_size=2)(net)

net = Conv2D(64, kernel_size=3, strides=1, padding='same', activation='relu')(net)
net = MaxPooling2D(pool_size=2)(net)

net = Conv2D(128, kernel_size=3, strides=1, padding='same', activation='relu')(net)
net = MaxPooling2D(pool_size=2)(net)

net = Flatten()(net)

net = Dense(512)(net)
net = Activation('relu')(net)
net = Dense(1)(net)
outputs = Activation('sigmoid')(net)

model = Model(inputs=inputs, outputs=outputs)

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['acc'])

model.summary()
```

설 계

눈 깜빡임 감지 머신러닝 소스코드

```
[6]
start_time = datetime.datetime.now().strftime('%Y_%m_%d_%H_%M_%S')

model.fit_generator(
    train_generator, epochs=50, validation_data=val_generator,
    callbacks=[
        ModelCheckpoint('C:/Users/skt/Desktop/Twinkle/%s.h5' % (start_time), monitor
='val_acc', save_best_only=True, mode='max', verbose=1),
        ReduceLROnPlateau(monitor='val_acc', factor=0.2, patience=10, verbose=1, mod
e='auto', min_lr=1e-05)
    ]
)
print("file name : "+start_time+".h5")
```

설 계

눈 깜빡임 감지 머신러닝 소스코드

```
[7]
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns

model = load_model('C:/Users/skt/Desktop/Twinkle/%s.h5' % (start_time))

y_pred = model.predict(x_val/255.)
y_pred_logical = (y_pred > 0.5).astype(np.int)

print ('test acc: %s' % accuracy_score(y_val, y_pred_logical))
cm = confusion_matrix(y_val, y_pred_logical)
sns.heatmap(cm, annot=True)


[8]
ax = sns.distplot(y_pred, kde=False)
```

설 계

머신러닝 소스코드 실행

In [10]:
```
start_time = datetime.datetime.now().strftime('%Y_%m_%d_%H_%M_%S')

model.fit_generator(
    train_generator, epochs=50, validation_data=val_generator,
    callbacks=[
        ModelCheckpoint('C:/Users/skt/Desktop/Twinkle/%s.h5' % (start_time), monitor='val_acc', save_best_only=True, mode='max', verbose
        ReduceLROnPlateau(monitor='val_acc', factor=0.2, patience=10, verbose=1, mode='auto', min_lr=1e-05)
    ]
)
print("file name : "+start_time+".h5")
```

```
81/81 [==============================] - ETA: 0s - loss: 0.0022 - acc: 0.9996
Epoch 00046: val_acc did not improve from 1.00000
81/81 [==============================] - 6s 73ms/step - loss: 0.0022 - acc: 0.9996 - val_loss: 0.0080 - val_acc: 0.9965
Epoch 47/50
81/81 [==============================] - ETA: 0s - loss: 0.0032 - acc: 0.9988
Epoch 00047: val_acc did not improve from 1.00000
81/81 [==============================] - 6s 74ms/step - loss: 0.0032 - acc: 0.9988 - val_loss: 0.0072 - val_acc: 0.9965
Epoch 48/50
81/81 [==============================] - ETA: 0s - loss: 0.0040 - acc: 0.9981
Epoch 00048: val_acc did not improve from 1.00000
81/81 [==============================] - 6s 73ms/step - loss: 0.0040 - acc: 0.9981 - val_loss: 0.0058 - val_acc: 0.9965
Epoch 49/50
81/81 [==============================] - ETA: 0s - loss: 0.0031 - acc: 0.9992
Epoch 00049: val_acc did not improve from 1.00000
81/81 [==============================] - 5s 67ms/step - loss: 0.0031 - acc: 0.9992 - val_loss: 0.0060 - val_acc: 0.9965
Epoch 50/50
81/81 [==============================] - ETA: 0s - loss: 0.0034 - acc: 0.9992
Epoch 00050: val_acc did not improve from 1.00000
81/81 [==============================] - 6s 71ms/step - loss: 0.0034 - acc: 0.9992 - val_loss: 0.0068 - val_acc: 0.9965
file name : 2020_12_07_19_04_40.h5
```

Jupyter notebook에 머신러닝 모듈파일 2020_12_07_19_04_40.h5 제작

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
import tensorflow.compat.v1 as tf
tf.disable_resource_variables()
tf.disable_v2_behavior()
import cv2, dlib
import numpy as np
from imutils import face_utils
from keras.models import load_model
import datetime
import time
import RPi.GPIO as GPIO

IMG_SIZE = (34, 26)

detector = dlib.get_frontal_face_detector()
predictor =
dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')
model = load_model('models/2020_12_07_19_04_40.h5')
model.summary()
GPIO.setmode(GPIO.BCM)          #led 사용을 위한 코드를 추가하였다.
GPIO.setwarnings(False)
GPIO.setup(18, GPIO.OUT)
```

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
pwm = GPIO.PWM(18, 50)          # PWM 18번 핀에 50Hz의 주파수 설정
pwm.start(100)        # PWM의 값을 0~100까지 넣을 수 있다. (100의 dutycicle 설정)
dc = 100                        - 100이 가장 밝고 0이 가장 어둡다.

fs = None               # 시간의 흐름을 알기 위해 추가하였다.
fm = None
count = 0
end loop = 0        # 자동 종료를 하기 위해 추가하였다.

def crop_eye(img, eye_points):
   x1, y1 = np.amin(eye_points, axis=0)
   x2, y2 = np.amax(eye_points, axis=0)
   cx, cy = (x1 + x2) / 2, (y1 + y2) / 2

w = (x2 - x1) * 1.2
h = w * IMG_SIZE[1] / IMG_SIZE[0]

margin_x, margin_y = w / 2, h / 2

min_x, min_y = int(cx - margin_x), int(cy - margin_y)

max_x, max_y = int(cx + margin_x), int(cy + margin_y)
```

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
eye_rect = np.rint([min_x, min_y, max_x, max_y]).astype(np.int)

eye_img = gray[eye_rect[1]:eye_rect[3], eye_rect[0]:eye_rect[2]]

return eye_img, eye_rect

# main
#cap = cv2.VideoCapture('videos/2.mp4')
 cap = cv2.VideoCapture(0)          # video 대신 Pi Camera를 실행한다.
#cap.set(3,640) # set Width
#cap.set(4,480) # set Height

while cap.isOpened():
  ret, img_ori = cap.read()

if not ret:
  break

img_ori = cv2.resize(img_ori, dsize=(0, 0), fx=0.5, fy=0.5)

img = img_ori.copy()
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
faces = detector(gray)

for face in faces:
    shapes = predictor(gray, face)
    shapes = face_utils.shape_to_np(shapes)

    eye_img_l, eye_rect_l = crop_eye(gray, eye_points=shapes[36:42])
    eye_img_r, eye_rect_r = crop_eye(gray, eye_points=shapes[42:48])

    eye_img_l = cv2.resize(eye_img_l, dsize=IMG_SIZE)
    eye_img_r = cv2.resize(eye_img_r, dsize=IMG_SIZE)
    eye_img_r = cv2.flip(eye_img_r, flipCode=1)

# cv2.imshow('l', eye_img_l)
# cv2.imshow('r', eye_img_r)

eye_input_l = eye_img_l.copy().reshape((1, IMG_SIZE[1], IMG_SIZE[0], 1)).astype(np.float32)
/ 255.
eye_input_r = eye_img_r.copy().reshape((1, IMG_SIZE[1], IMG_SIZE[0], 1)).astype(np.float32)
/ 255.

pred_l = model.predict(eye_input_l)
pred_r = model.predict(eye_input_r)
```

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
# visualize
state_l= '0 %.1f' if pred_l > 0.1 else '- %.1f'
      #출력되는 동영상에서 눈 부분의 네모 창 위에 떠있는 숫자이다. pred_l, pred_r이 0.1보다 작은 경우 -가 앞에 붙는다.
       따라서 pred_l이 0.1보다 작으면 눈을 감았다고 예측한다.
state_r = '0 %.1f' if pred_r > 0.1 else '- %.1f'

state_l = state_l % pred_l
state_r = state_r % pred_r

cv2.rectangle(img, pt1=tuple(eye_rect_l[0:2]), pt2=tuple(eye_rect_l[2:4]),
color=(255,255,255), thickness=2)
cv2.rectangle(img, pt1=tuple(eye_rect_r[0:2]), pt2=tuple(eye_rect_r[2:4]),
 color=(255,255,255), thickness=2)

cv2.putText(img, state_l, tuple(eye_rect_l[0:2]), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
(255,255,255), 2)
cv2.putText(img, state_r, tuple(eye_rect_r[0:2]), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
(255,255,255), 2)
```

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
#눈을 감았을때의 초(시간)를 세는 코드
if pred_l < 0.1 and pred_r < 0.1 :
    if fs is None :      #양쪽 눈을 모두 감았을 때 fs값이 None이면 이 코드를 실행한다.
        fnow = datetime.datetime.now()        #datetime클래스를 사용해서 오늘의 날짜와 시간값을 fnow에 저장한다.
        fs = int(fnow.strftime('%S'))  // second
        fm = int(fnow.strftime('%M'))  // minute

    else :
        snow = datetime.datetime.now()
        ss = int(snow.strftime('%S'))
        sm = int(snow.strftime('%M'))

        if ss == 0 or ss == 60 :     #입력 받은 초가 0 또는 60초이면

            count = count + 1      #count 변수에 1을 더한다.

            sscount = ss + (count*60)      #0또는 60초가 반복된 횟수 count 변수에 60을 곱한 값을 더한 것이다.

        else :        #그 이외의 경우
            sscount = ss     #숫자 그대로 사용한다.
```

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
#10초 후에 LED가 꺼짐
if (sscount - fs) < 10 :          #지나간 초가 10초보다 작을 경우
    pwm.ChangeDutyCycle(dc)  # dc = 100~0까지의 값을 설정할 수 있고  100에서 1초당 10씩 10번 감소하면 0이다.
    dc = 100 -(sscount - fs)*10      #10초 동안 led가 서서히 꺼진다.

else :
    pwm.stop()
    print("LED OFF")
    endloop = 1
    break

#10분 후에 LED가 꺼짐
#if sm == 0 or sm == 60 :
    # count = count + 1
    # smcount = sm + (count*60)

#else :
    # smcount = sm

#if (smcount - fm) < 10 :    #10분
   # pwm.ChangeDutyCycle(dc)

#if (sscount-fs)%10 == 0 :              #led 밝기가 약 10초에 1씩 감소한다.
   # dc = dc - 1
```

설 계

눈 깜빡임 응용 에너지 절약 장치 소스코드

```
#else :       #10분 이상인 경우
    # pwm.stop()      #led를 끈다.
    # print("LED OFF")
    # endloop = 1
    # break
print("closeing eye time :" + str(sscount-fs))

else :
    #pred_l과 pred_r이 0.1보다 클 때 실행된다. 눈을 감았다는 것을 인지하기 전의 초기 설정으로 리셋시킬 필요가 있다.
     fs = None
     fm=None
     dc = 100
     pwm.ChangeDutyCycle(dc)
     print("Awke")     #눈을 떴다는 것을 터미널 창에 알려준다.

cv2.imshow('result', img)

if endloop == 1 :
   break                # 프로그램을 자동으로 종료해준다.

if cv2.waitKey(1) == ord('q'):
     break

cap.release()
```

INDEX

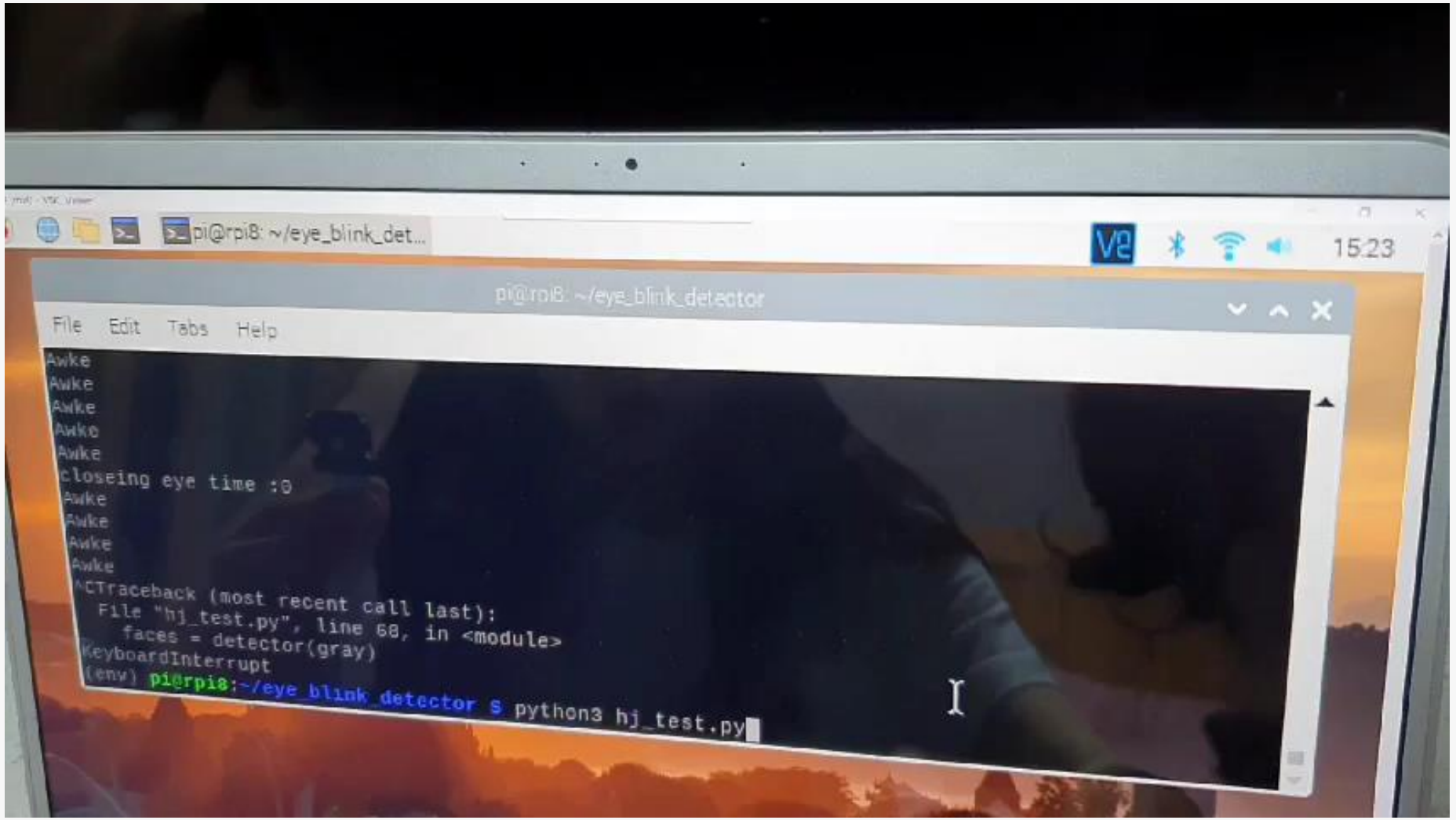
3 결론

팀 성과 (결과), 개선 사항

느낀점 및 알게 된 점

결 론

팀 성과 (결과)



결 론

팀 성과, 개선 사항

1. TensorFlow 설치 오류

wget https://github.com/lhelontra/tensorflow-on-arm/releases/download/v2.0.0/tensorflow-2.0.0-cp37-none-linux_armv7l.whl
python3 -m pip uninstall tensorflow
python3 -m pip install tensorflow-2.0.0-cp37-none-linux_armv7l.whl <- TensorFlow 2.0.0로 설치하여 오류 해결

2. Keras 설치 오류

python -m pip install Keras==2.3.1 <- Keras를 2.3.1로 업그레이드
import tensorflow.compat.v1 as tf
tf.disable_v2_behavior() 문구 추가하여 해결

결 론

느낀점 및 알게 된 점

[김하얀]
이번 설계를 통해 직접 실습을 해 보니 Raspberry Pi를 이용한 센서모듈(카메라, LED 등) 연결,
GitHub, Jupyter Notebook 등의 Raspberry Pi를 다루는 방법을 알게 되었고 아직 Raspberry Pi에 대해
모르는 부분이 많다고 느꼈다.

[김현정]
실습을 통해 센서를 사용하기 위한 소스코드 이해 및 실제 센서로부터 입력받는 값들을 활용하는 방법을 알게 되었다.
오픈 소스를 활용한 실습을 통해 머신러닝 분석 데이터와 소스코드는 다양한 활용이 가능하다는 것을 알게 되었다.

[맹하늘]
작품설계를 통해 라즈베리 파이를 실습한 내용들에 대한 이해도가 향상되었다. 또한, 머신러닝 오픈 소스를 통해
머신러닝에 대해 알게되었고 조금이나마 응용할 수 있게 되었다.

[신선영]
라즈베리 파이에 보드를 연결해 간단한 설계도 해보고, 주제에 맞는 코드를 입력했을 때 원하는 값이 나오지 않고
에러가 출력되는 경우가 더 많았다. 원인이 무엇인지 찾아보고, 다시 코드를 실행시키는 과정을 여러번 반복해보며
원하는 결과값을 얻었을 때 어느때보다도 성취감과 협력의 중요함을 많이 느낄 수 있는 시간이었다.

결 론

느낀점 및 알게 된 점

[장하영]
Windows 환경에서 짜인 오픈소스를 Raspberry pi에서 구현하다보니 TensorFlow, Keras, OpenCV 등의 초기 개발 환경을 구축할 때 오류가 잦았다. 오류의 원인이 모듈 버전 문제임을 알아내어 가상환경을 구성하고 오류를 해결해 나갔다. 해결한 오류들을 팀원들과 공유할 땐 서로 도움이 됐고 힘이 났다. 우리의 아이디어를 실제로 구현해내는데에 성공했을 땐 신기하고 뿌듯했다. 앞으로의 개발활동에 자신감이 생겼고, 추가로 아이디어를 내어 더 응용해보고 싶다.

[최유진]
설계 주제를 고를때 예제들을 보면서 Raspberry Pi를 이용한 다양한 머신러닝 응용이 가능하다는 것을 알았다.
오픈소스 실행 과정에서 많은 오류들이 발생하였지만 서로 협력하여 하나하나 문제를 해결하던 점에서 팀워크를 느낄 수 있었다. 실제로 머신러닝 실습을 진행하면서 Raspberry Pi와 기타 프로그램들의 사용법을 좀 더 폭넓게 익히게 되었다.

INDEX

마 무 리

바라는 [교과 연계형 비교과 프로그램]

직접 실습하고 작품을 설계하여 교과 프로그램 (AI 머신러닝, 딥러닝등) 에 대한 이해도를 높일수 있는 실질적인 과제형 프로그램

마 무 리

참고문헌

빵형의 개발도상국 - 딥러닝으로 눈 깜빡임 감지기 만들기
https://youtu.be/dJjzTo8_x3c

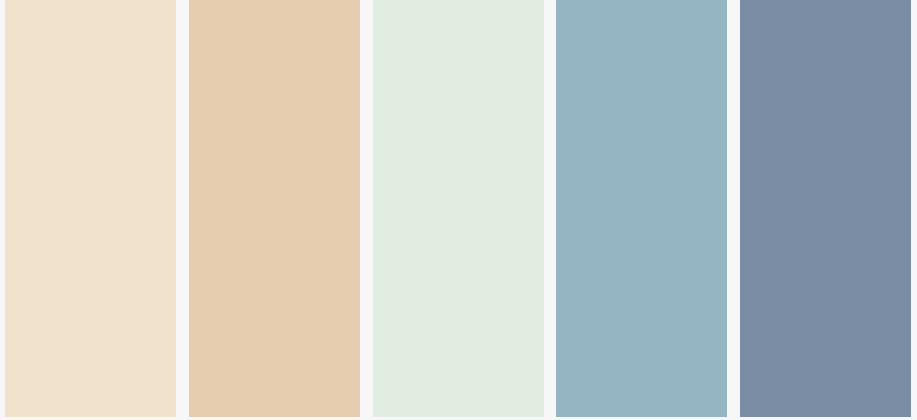
눈 깜빡임 소스코드(test.py)
https://github.com/kairess/eye_blink_detector

머신러닝 정의
https://terms.naver.com/entry.nhn?docId=3347329&cid=40942&categoryId=32845
https://m.blog.naver.com/sbd38/221369711732

텐서플로우 정의
https://terms.naver.com/entry.nhn?docId=3434677&cid=40942&categoryId=32837

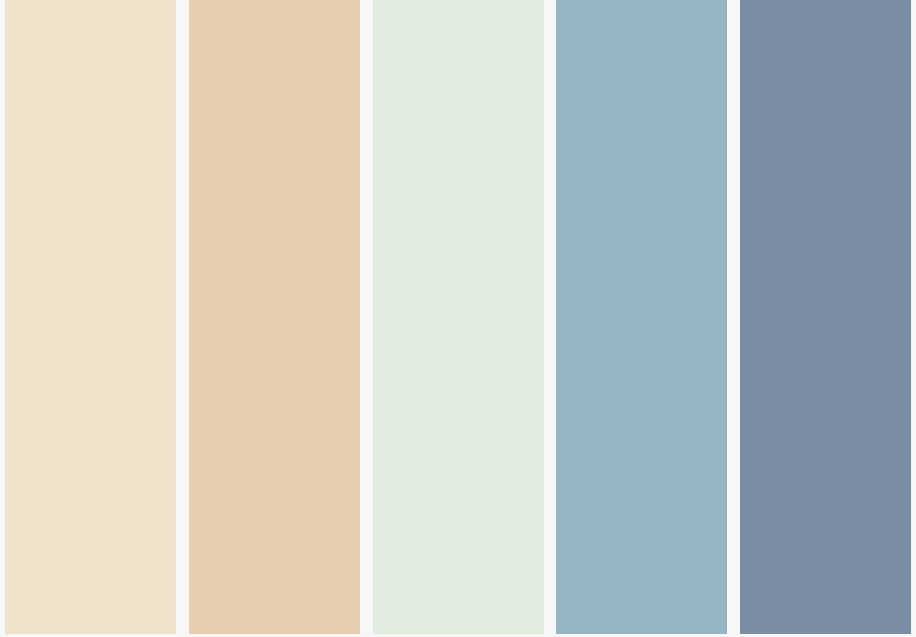
케라스 정의
https://ko.wikipedia.org/wiki/%EC%BC%80%EB%9D%BC%EC%8A%A4
https://blog.naver.com/beyondlegend/222141341172

사물인터넷을 위한 리눅스 프로그래밍 with 라즈베리 파이



?

QUESTION & ANSWER



THANK YOU!