RECEP TAYYIP ERDOGAN ÜNİVERSİTESİ
MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

INTERNET OF THINGS

2023-2024 BAHAR DÖNEMİ

Final Projesi

ESP32 Gömülü Cihazına Cifar10 CNN Modelinin Gömülmesi ve Çalıştırılması

Teslim Tarihi: 10 Haziran 2024

Mehmet KADIOĞLU

201401033

Öğretim Görevlisi

Dr. Öğr. Üyesi Yıldıran YILMAZ

RIZE

2024

İçindekiler

1. Giriş

Günümüzde küçük hafıza ve işlem gücüne sahip cihazlar gittikçe artmaktadır. Bunlar arasında akıllı saat, mobil telefon, mikrodenetleyiciler, akıllı ev aletleri, televizyon, giyilebilir teknolojiler vb. bulunmaktadır.

Bu projede küçük hafızaya ve işlem gücüne sahip mikrodenetleyiciler kullanılarak, bu cihazların üzerinde mikrodenetleyicilere özel olarak optimize edilmiş resim tanımlama ve sınıflandırma yapay zeka modellerinin çalıştırılması planlanmaktadır. Bu şekilde çok fazla işlem gücü, hafıza ve güç gerektirmeyen taşınabilir cihazlar üzerinde obje sınıflandırılması yapılması planlanmaktadır.

Bilgisayar veya pahalı yüksek işlem gücü ve hafızaya sahip cihazlar yerine; hem maliyet olarak ucuz hemde basit bir batarya ile uzun süreler çalışabilecek bir görüntü sınıflandırma cihazı yapılması ve akıllı ev, iş yeri vb. alanlarda kullanılması planlanmaktadır.

2. Proje Tanımı ve Amacı

Bu projede kedi ve köpek olmak üzere 2 tane resim sınıfı kullanılarak espressif [1] cihazlar üzerinde resim sınıflandırılması yapmak üzere çalıştırılması planlanmaktadır. cifar10 [2] veri seti içindeki kedi ve köpek sınıflarına ait resimlerin kullanılması planlanmaktadır. 2 sınıfa ait toplam 12.000 resim için bir yapay zeka modeli oluşturulması ve bu modelin espressif cihaza gömülmesi hedeflenmektedir. Böylece cihaza gömülen model ile hem taşınabilir olması hemde ucuz olması bakımından akıllı ev projelerinde kullanılması planlanmaktadır.

3. Yöntem ve Kullanılan Malzemeler veya Gereksinimler

3.1 Kullanılan Malzemeler ve Araçlar

Kullanılan Yazılımsal Araçlar:

Kullanılması kolay ve gelişmiş araçlara sahip olduğu için tercih edildi.

- PlatformIO

PlatformIO serial ekranı çalışmadığı için Ardunio IDE serial ekranı kullanıldı.

- Serial Ekranın Görüntülenebilmesi için Arduino IDE Serial Port Ekranı

TensorFlow Lite ve keras kütüphaneleri ile python3 üzerinde yazılan model beklenildiği gibi çalışmadığı için Edge Impulse üzerinden eğitilmiş model alındı ve kütüphane olarak kullanıldı.

- Edge Impulse
- TensorFlow Lite
- Python3

- Keras kütüphanesi

Kullanılan Donanımsal Araçlar:

OTA özelliğinin kullanılmaması durumunda 3MB'a kadar flash hafızası ve 512KB SRAM hafızası olduğu için optimize edilmiş yapay zeka modelini çalıştırabilir [3].

- ESP32S mikrodenetleyici

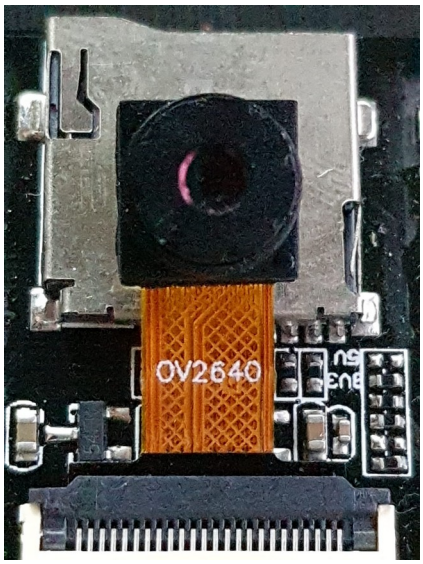
Üzerinde OV2640 kamera ile gelen ESP32CAM modülü kullanıldı.

- ESP32CAM modülü
- OV2640 Kamera

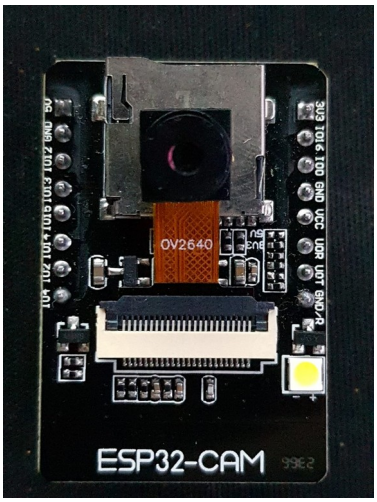
ESP32CAM'e micro usb portu üzerinden kod yükenebilmesi için ve pinler kullanılmadan reset atılabilmesi için kullanıldı.

- ESP32CAM modülü kod yükleme dönüştürücüsü ve reset butonu

3.2 Devre Tasarımı ve Şeması

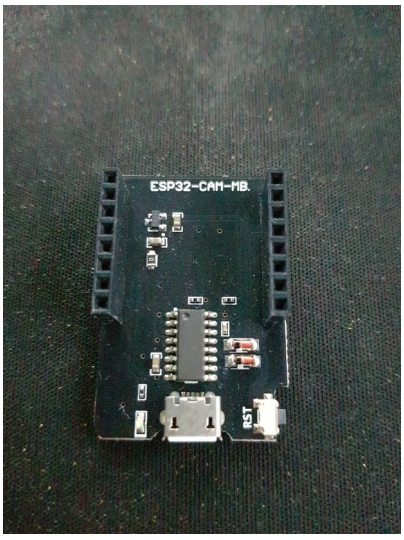


Şekil 1 - OV2640 Kamera



Şekil 2 - ESP32CAM modülü

Şekil 3 - ESP32S mikrodenetleyici



Şekil 4 - ESP32CAM kod yükleme dönüştürücüsü

4. Arduino Kodları ve Açıklamaları

İlk önce ilgili websitesindeki cifar10 projesi klon edildi [4]. Daha sonra Edge Impulse[5] üzerinden ilgili modelin çıktısının alınabilmesi için “deploy” edildi. İşlem bittikten sonra deploy edilen kütüphane .zip formatında indirildi. PlatformIO projesine, çıktı alınan kütüphanenin eklenebilmesi için aşağıdaki kod çalıştırıldı:

“~/.platformio/penv/bin/pio lib install lib/cifar10_final-v3.zip”

4.1 Kod Parçası 1

main.cpp

```
#include <Cifar10_final_inferencing.h>

#include "edge-impulse-sdk/dsp/image/image.hpp"

#include "esp_camera.h"
```

İlk önce ilgili kütüphaneler import edildi.

4.2 Kod Parçası 2

main.cpp

```
//#define CAMERA_MODEL_ESP_EYE // Has PSRAM

#define CAMERA_MODEL_AI_THINKER // Has PSRAM

#if defined(CAMERA_MODEL_ESP_EYE)
#define PWDN_GPIO_NUM -1
#define RESET_GPIO_NUM -1
#define XCLK_GPIO_NUM 4
#define SIOD_GPIO_NUM 18
#define SIOC_GPIO_NUM 23

#define Y9_GPIO_NUM 36
#define Y8_GPIO_NUM 37
#define Y7_GPIO_NUM 38
#define Y6_GPIO_NUM 39
#define Y5_GPIO_NUM 35
#define Y4_GPIO_NUM 14
#define Y3_GPIO_NUM 13
#define Y2_GPIO_NUM 34
#define VSYNC_GPIO_NUM 5
#define HREF_GPIO_NUM 27
#define PCLK_GPIO_NUM 25

#elif defined(CAMERA_MODEL_AI_THINKER)
```

```
#define PWDN_GPIO_NUM 32
#define RESET_GPIO_NUM -1
#define XCLK_GPIO_NUM 0
#define SIOD_GPIO_NUM 26
#define SIOC_GPIO_NUM 27

#define Y9_GPIO_NUM 35
#define Y8_GPIO_NUM 34
#define Y7_GPIO_NUM 39
#define Y6_GPIO_NUM 36
#define Y5_GPIO_NUM 21
#define Y4_GPIO_NUM 19
#define Y3_GPIO_NUM 18
#define Y2_GPIO_NUM 5
#define VSYNC_GPIO_NUM 25
#define HREF_GPIO_NUM 23
#define PCLK_GPIO_NUM 22

#else
#error "Camera model not selected"
#endif
```

Daha sonra ESP-EYE ve ESP-AI-THINKER için pin tanımlamaları yapıldı.

4.3 Kod Parçası 3

main.cpp

```
/* Constant defines ------------------------------------------------------- */

#define EI_CAMERA_RAW_FRAME_BUFFER_COLS 320
#define EI_CAMERA_RAW_FRAME_BUFFER_ROWS 240
#define EI_CAMERA_FRAME_BYTE_SIZE 3

/* Private variables ------------------------------------------------------- */
static bool debug_nn = false; // Set this to true to see e.g. features generated from the raw signal
static bool is_initialised = false;
uint8_t *snapshot_buf; //points to the output of the capture

static camera_config_t camera_config = {
.pin_pwdn = PWDN_GPIO_NUM,
.pin_reset = RESET_GPIO_NUM,
.pin_xclk = XCLK_GPIO_NUM,
.pin_sscb_sda = SIOD_GPIO_NUM,
.pin_sscb_scl = SIOC_GPIO_NUM,

.pin_d7 = Y9_GPIO_NUM,
```

```
.pin_d6 = Y8_GPIO_NUM,
.pin_d5 = Y7_GPIO_NUM,
.pin_d4 = Y6_GPIO_NUM,
.pin_d3 = Y5_GPIO_NUM,
.pin_d2 = Y4_GPIO_NUM,
.pin_d1 = Y3_GPIO_NUM,
.pin_d0 = Y2_GPIO_NUM,
.pin_vsync = VSYNC_GPIO_NUM,
.pin_href = HREF_GPIO_NUM,
.pin_pclk = PCLK_GPIO_NUM,

//XCLK 20MHz or 10MHz for OV2640 double FPS (Experimental)
.xclk_freq_hz = 20000000,
.ledc_timer = LEDC_TIMER_0,
.ledc_channel = LEDC_CHANNEL_0,

.pixel_format = PIXFORMAT_JPEG, //YUV422,GRAYSCALE,RGB565,JPEG
.frame_size = FRAMESIZE_QVGA, //QQVGA-UXGA Do not use sizes above QVGA when not JPEG

.jpeg_quality = 12, //0-63 lower number means higher quality
.fb_count = 1, //if more than one, i2s runs in continuous mode. Use only with JPEG
.fb_location = CAMERA_FB_IN_PSRAM,
.grab_mode = CAMERA_GRAB_WHEN_EMPTY,
};

/* Function definitions ------------------------------------------------------- */
bool ei_camera_init(void);
void ei_camera_deinit(void);
bool ei_camera_capture(uint32_t img_width, uint32_t img_height, uint8_t *out_buf) ;
static int ei_camera_get_data(size_t offset, size_t length, float *out_ptr);
```

Daha sonra Edge impulse kütüphanesi ile ilgili bazı sabitler ve değişkenler tanımlandı. Kamera için ilgili pinler tanımlandı ve fonksiyon tanımlamaları yapıldı.

4.4 Kod Parçası 4

main.cpp

```
void setup()

{
// put your setup code here, to run once:
Serial.begin(115200);
//comment out the below line to start inference immediately after upload
while (!Serial);
Serial.println("Edge Impulse Inferencing Demo");
if (ei_camera_init() == false) {
ei_printf("Failed to initialize Camera!\r\n");
}
```

```
else {
ei_printf("Camera initialized\r\n");
}

ei_printf("\nStarting continious inference in 2 seconds...\n");
ei_sleep(2000);
}
```

Setup fonksiyonunda ilk önce serial ekran başlatıldı daha sonra ise kamera başlatılması denendi. Daha sonra loopa geçmeden önce 2 saniye beklendi.

4.5 Kod Parçası 5

main.cpp

```
void loop()

{

// instead of wait_ms, we'll wait on the signal, this allows threads to cancel us...
if (ei_sleep(5) != EI_IMPULSE_OK) {
return;
}

snapshot_buf = (uint8_t*)malloc(EI_CAMERA_RAW_FRAME_BUFFER_COLS *
EI_CAMERA_RAW_FRAME_BUFFER_ROWS * EI_CAMERA_FRAME_BYTE_SIZE);

// check if allocation was successful
if(snapshot_buf == nullptr) {
ei_printf("ERR: Failed to allocate snapshot buffer!\n");
return;
}

ei::signal_t signal;
signal.total_length = EI_CLASSIFIER_INPUT_WIDTH * EI_CLASSIFIER_INPUT_HEIGHT;
signal.get_data = &ei_camera_get_data;

if (ei_camera_capture((size_t)EI_CLASSIFIER_INPUT_WIDTH,
(size_t)EI_CLASSIFIER_INPUT_HEIGHT, snapshot_buf) == false) {
ei_printf("Failed to capture image\r\n");
free(snapshot_buf);
return;
}

// Run the classifier
ei_impulse_result_t result = { 0 };

EI_IMPULSE_ERROR err = run_classifier(&signal, &result, debug_nn);
if (err != EI_IMPULSE_OK) {
ei_printf("ERR: Failed to run classifier (%d)\n", err);
```

```
return;
}

// print the predictions
ei_printf("Predictions (DSP: %d ms., Classification: %d ms., Anomaly: %d ms.): \n",
result.timing.dsp, result.timing.classification, result.timing.anomaly);

#if EI_CLASSIFIER_OBJECT_DETECTION == 1
ei_printf("Object detection bounding boxes:\r\n");
for (uint32_t i = 0; i < result.bounding_boxes_count; i++) {
ei_impulse_result_bounding_box_t bb = result.bounding_boxes[i];
if (bb.value == 0) {
continue;
}
ei_printf(" %s (%f) [ x: %u, y: %u, width: %u, height: %u ]\r\n",
bb.label,
bb.value,
bb.x,
bb.y,
bb.width,
bb.height);
}

// Print the prediction results (classification)
#else
ei_printf("Predictions:\r\n");
for (uint16_t i = 0; i < EI_CLASSIFIER_LABEL_COUNT; i++) {
ei_printf(" %s: ", ei_classifier_inferencing_categories[i]);
ei_printf("%.5f\r\n", result.classification[i].value);
}
#endif

// Print anomaly result (if it exists)
#if EI_CLASSIFIER_HAS_ANOMALY
ei_printf("Anomaly prediction: %.3f\r\n", result.anomaly);
#endif

#if EI_CLASSIFIER_HAS_VISUAL_ANOMALY
ei_printf("Visual anomalies:\r\n");
for (uint32_t i = 0; i < result.visual_ad_count; i++) {
ei_impulse_result_bounding_box_t bb = result.visual_ad_grid_cells[i];
if (bb.value == 0) {
continue;
}
ei_printf(" %s (%f) [ x: %u, y: %u, width: %u, height: %u ]\r\n",
bb.label,
bb.value,
bb.x,
bb.y,
```

```
bb.width,
bb.height);
}
#endif


free(snapshot_buf);
}
```

Loopun içinde ilk önce buffer oluşturup kameradan gelen görüntü bu buffera atandı daha sonra ilgili resim sınıflandırıcıya sokularak, sınıf label'larına göre ilgili tahminler serial ekrana bastırıldı. Eğer bir anomali varsa onuda bastıracak şekilde ayarlandı.

4.6 Kod Parçası 6

main.cpp

```
bool ei_camera_init(void) {

if (is_initialised) return true;

#if defined(CAMERA_MODEL_ESP_EYE)
pinMode(13, INPUT_PULLUP);
pinMode(14, INPUT_PULLUP);
#endif

//initialize the camera
esp_err_t err = esp_camera_init(&camera_config);
if (err != ESP_OK) {
Serial.printf("Camera init failed with error 0x%x\n", err);
return false;
}

sensor_t * s = esp_camera_sensor_get();
// initial sensors are flipped vertically and colors are a bit saturated
if (s->id.PID == OV3660_PID) {
s->set_vflip(s, 1); // flip it back
s->set_brightness(s, 1); // up the brightness just a bit
s->set_saturation(s, 0); // lower the saturation
}

#if defined(CAMERA_MODEL_M5STACK_WIDE)
s->set_vflip(s, 1);
s->set_hmirror(s, 1);
#elif defined(CAMERA_MODEL_ESP_EYE)
s->set_vflip(s, 1);
s->set_hmirror(s, 1);
s->set_awb_gain(s, 1);
#endif
```

```
is_initialised = true;
return true;
}


void ei_camera_deinit(void) {

//deinitialize the camera
esp_err_t err = esp_camera_deinit();

if (err != ESP_OK)
{
ei_printf("Camera deinit failed\n");
return;
}

is_initialised = false;
return;
}
```

Gömülü cihaz kamerasını başlatmak ve devre dışı bırakmak için edge impulse wrapper fonksiyonları eklendi.

4.7 Kod Parçası 7

main.cpp

```
bool ei_camera_capture(uint32_t img_width, uint32_t img_height, uint8_t *out_buf) {

bool do_resize = false;

if (!is_initialised) {
ei_printf("ERR: Camera is not initialized\r\n");
return false;
}

camera_fb_t *fb = esp_camera_fb_get();

if (!fb) {
ei_printf("Camera capture failed\n");
return false;
}

bool converted = fmt2rgb888(fb->buf, fb->len, PIXFORMAT_JPEG, snapshot_buf);

esp_camera_fb_return(fb);

if(!converted){
ei_printf("Conversion failed\n");
```

```
return false;
}

if ((img_width != EI_CAMERA_RAW_FRAME_BUFFER_COLS)
|| (img_height != EI_CAMERA_RAW_FRAME_BUFFER_ROWS)) {
do_resize = true;
}

if (do_resize) {
ei::image::processing::crop_and_interpolate_rgb888(
out_buf,
EI_CAMERA_RAW_FRAME_BUFFER_COLS,
EI_CAMERA_RAW_FRAME_BUFFER_ROWS,
out_buf,
img_width,
img_height);
}


return true;
}
```

Gömülü cihazın yakaladığı resmi buffer'a aldıktan sonra buffer'dan RGB'ye dönüştürür. Eğer yakalanan resim belirlenen sabitler ile aynı değerde değil ise yeniden boyutu ayarlanır ve kırpılır.

4.8 Kod Parçası 8

main.cpp

```
static int ei_camera_get_data(size_t offset, size_t length, float *out_ptr)

{
// we already have a RGB888 buffer, so recalculate offset into pixel index
size_t pixel_ix = offset * 3;
size_t pixels_left = length;
size_t out_ptr_ix = 0;

while (pixels_left != 0) {
// Swap BGR to RGB here
// due to https://github.com/espressif/esp32-camera/issues/379
out_ptr[out_ptr_ix] = (snapshot_buf[pixel_ix + 2] << 16) + (snapshot_buf[pixel_ix +
1] << 8) + snapshot_buf[pixel_ix];

// go to the next pixel
out_ptr_ix++;
pixel_ix+=3;
pixels_left--;
}
// and done!
```

```
return 0;
}
```

Buffer’daki resmi BGR formatından RGB formatına dönüştürür.

5. Sonuçlar ve Tartışma

Projede kişisel olarak oluşturulan model kullanılamadı çünkü doğruluk oranı full-integer quantization işleminden sonra aşırı derecede düşüş gösterdi. Bundan dolayı edge impulse üzerinden model oluşturuldu. Projenin asıl hedefine ulaşılmakla beraber cifar 10 veri seti üzerinde var olan 10 tane sınıfın hepsi kullanılamadı. Fazla yer kaplayabileceğinden ötürü 2 tane sınıf kullanıldı ve test işlemi gerçekleştirildi. Test sonucunda çalışan kodun köpek ve kedileri doğru bir şekilde ayırt edebildiği gözlendi.

6. Sonuç

Mevcut olarak oluşturulan modelde kedi ve köpek olmak üzere 2 tane sınıf bulunmaktadır. Bu sınıfların sayısı gelecekte geliştirme yapılması planlanırsa arttırılmaya çalışılabilir. Edge impulse üzerinden yapmak yerine kişisel model oluşturmak adına, full-integer quantization işlemi için başka bir yöntem denenebilir ve doğruluk oranının düşüşü azaltılmaya çalışılabilir. Eğer bu yöntem de çalışmaz ise keras kütüphanesi yerine başka bir kütüphane ile yeni bir model eğitilmesi denenebilir.

7. Ekler

Devre şeması görüntüsü, proje kodları ve README dosyasındaki açıklamalara aşağıdaki linkten bakılabilir:

https://github.com/mehmet-kadioglu/cifar10-on-esp32cam/tree/main/esp32cam-cifar10-recognition

8. Kaynaklar

[1] “Wi-Fi & Bluetooth MCUs and AIoT Solutions I Espressif Systems,” www.espressif.com. https://www.espressif.com/

[2] A. Krizhevsky, “CIFAR-10 and CIFAR-100 datasets,” Toronto.edu, 2009. https://www.cs.toronto.edu/~kriz/cifar.html

[3] “ESP32S Data Sheet” Ai-thinker.com, 2024. https://docs.ai-thinker.com/en/esp32/spec/esp32s

[4] “MJRoBot (Marcelo Rovai) / Cifar10_Image_Classification - Data acquisition,” studio.edgeimpulse.com. https://studio.edgeimpulse.com/public/51070/latest/acquisition/training.

[5] “Edge Impulse,” edgeimpulse.com. https://edgeimpulse.com/