



OpenCV China

SHENZHEN INSTITUTE

of ARTIFICIAL INTELLIGENCE AND ROBOTICS *for* SOCIETY

深圳市人工智能与机器人研究院

# Develop A Face Recognition System on ARM Board Using OpenCV

WU Jia

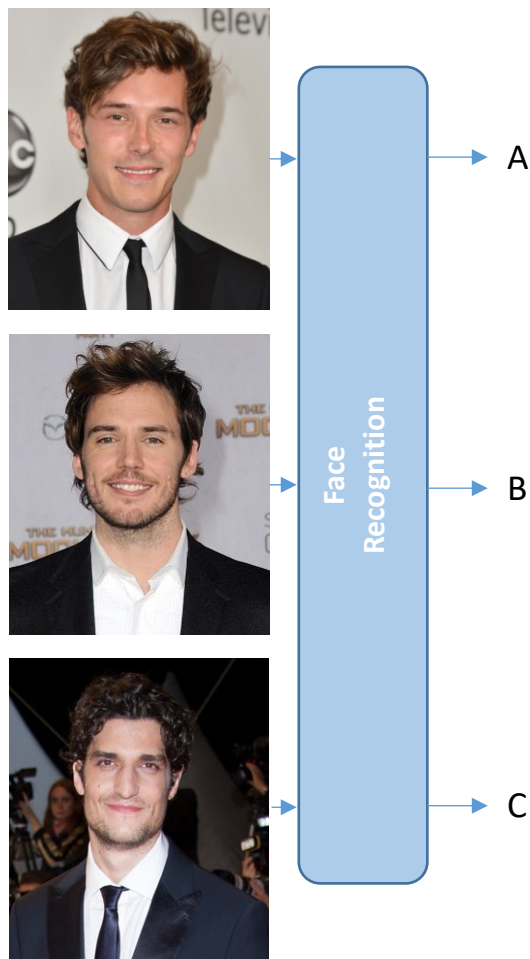
OpenCV China Team



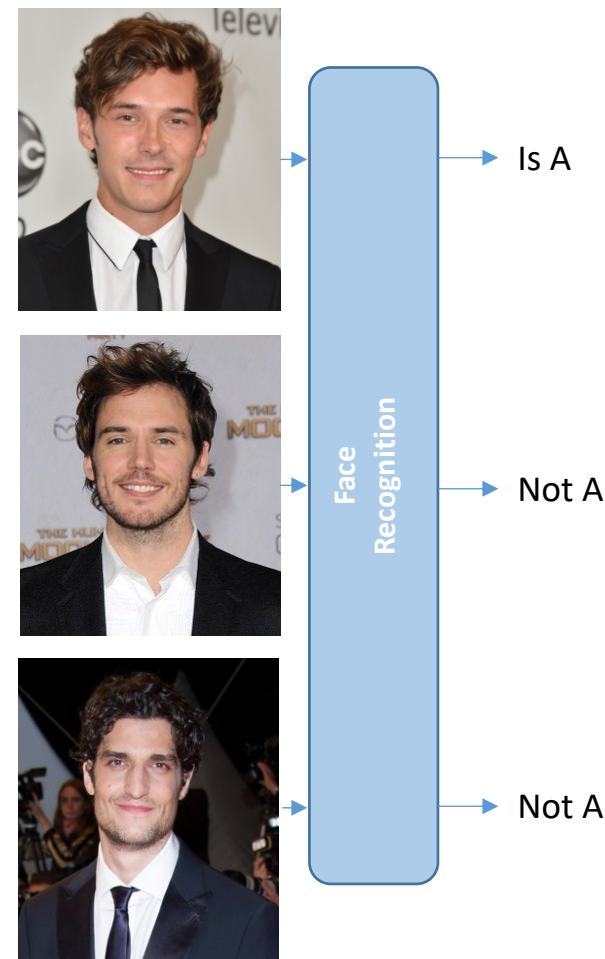
# Outline

- Face recognition in brief
- Build a FR system on ARM board using OpenCV
- Assignment

# Face recognition is to identify or verify a person from an image or a video frame.

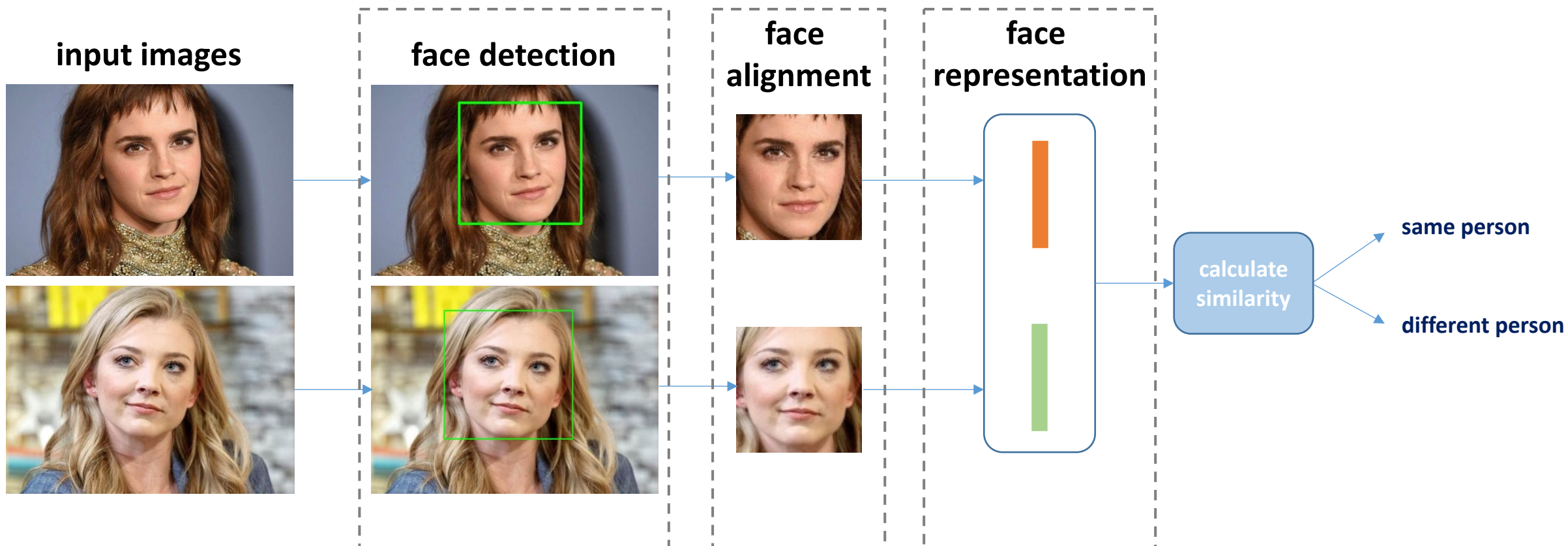


Who is this person, 1:N



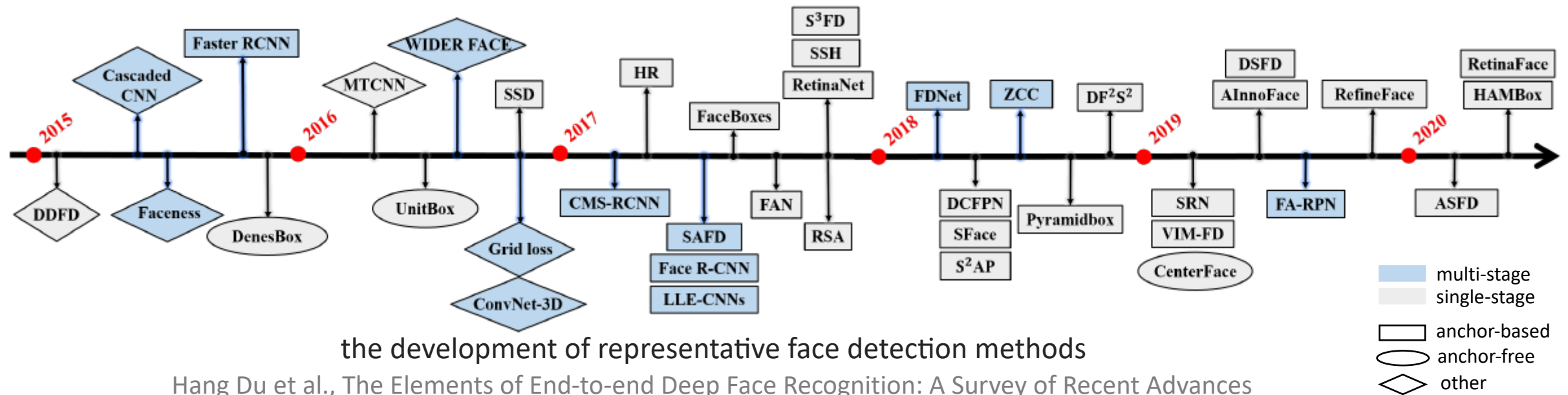
Is this person X, 1:1

# Face Recognition Pipeline



# Face Detection

- Traditional methods  
Viola-Jones face detector, DPM etc.
- Deep learning based



the development of representative face detection methods

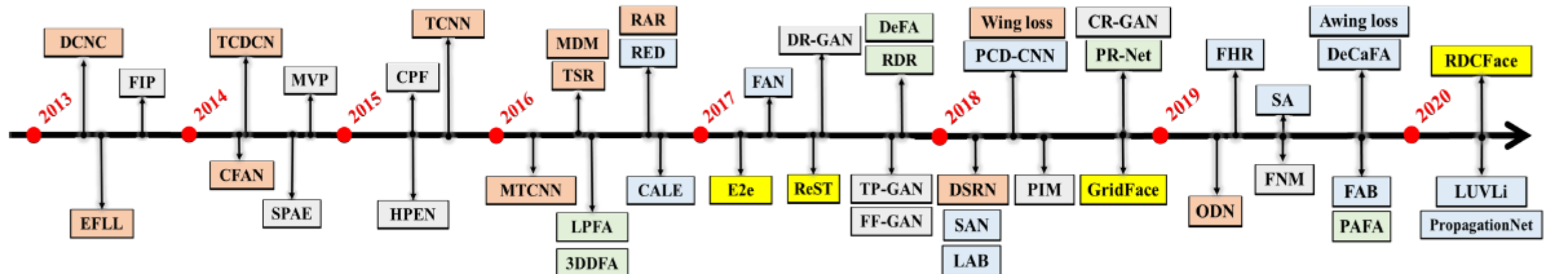
Hang Du et al., The Elements of End-to-end Deep Face Recognition: A Survey of Recent Advances

# Face Alignment

To obtain a canonical layout of the unconstrained face.



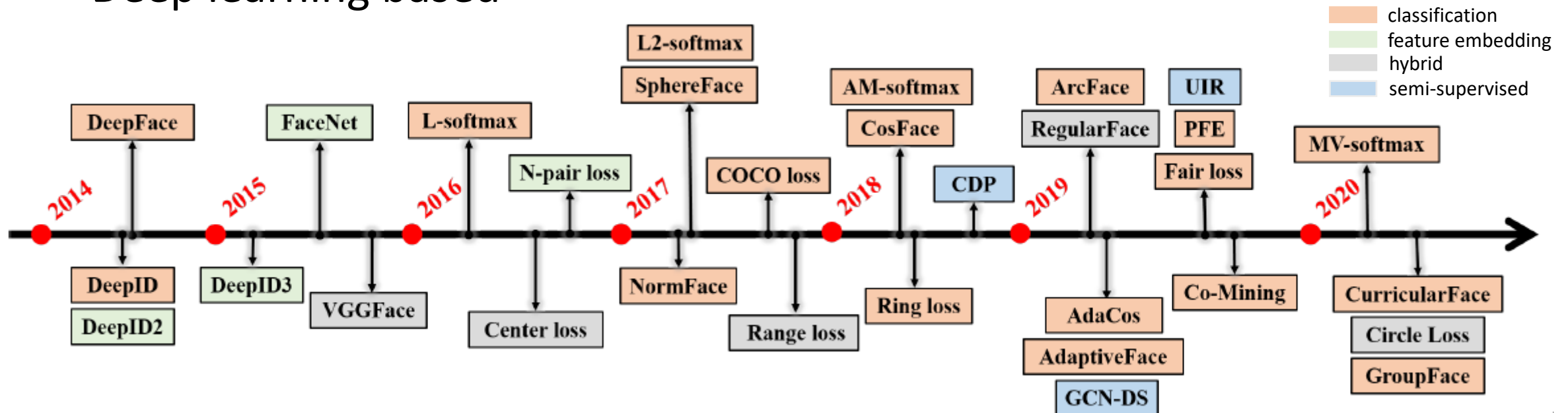
- Landmark-based methods
- Landmark-free methods
- Face frontalization



# Face Representation

To extract discriminative features from aligned face images, which is key to a face recognition system.

- Traditional methods  
Eigenface, Gabor, LBP etc.
- Deep learning based



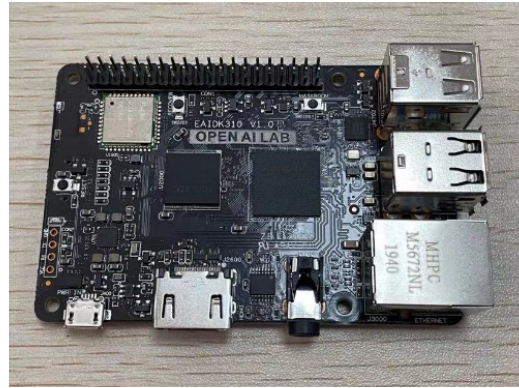


# Outline

- Face recognition in brief
- Build a FR system on ARM board using OpenCV
- Assignment



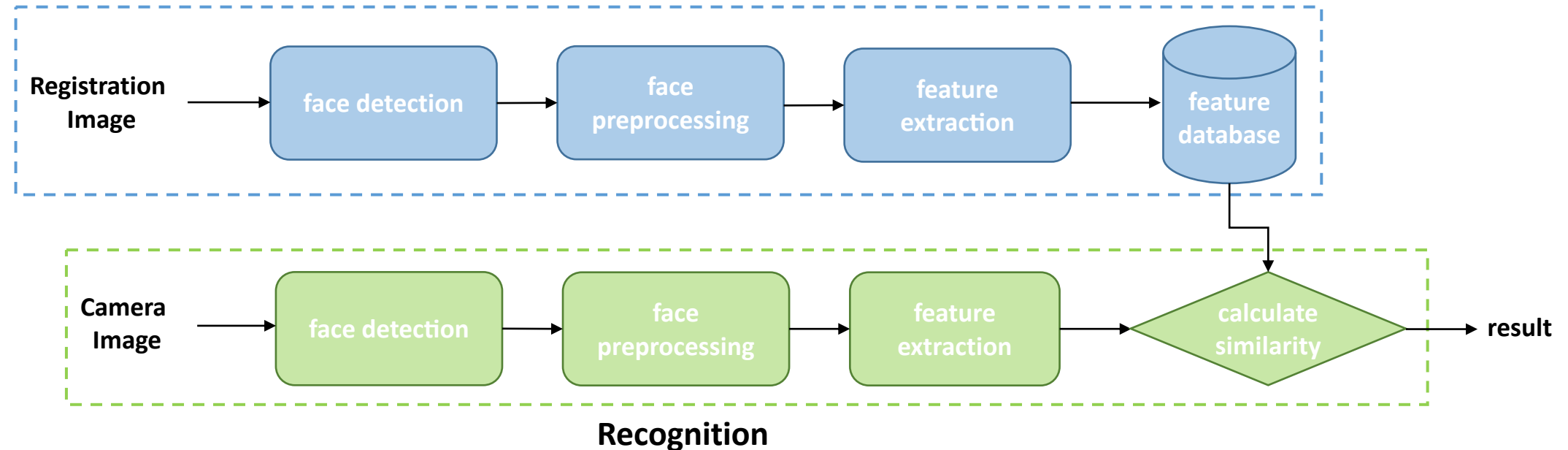
ARM dev board



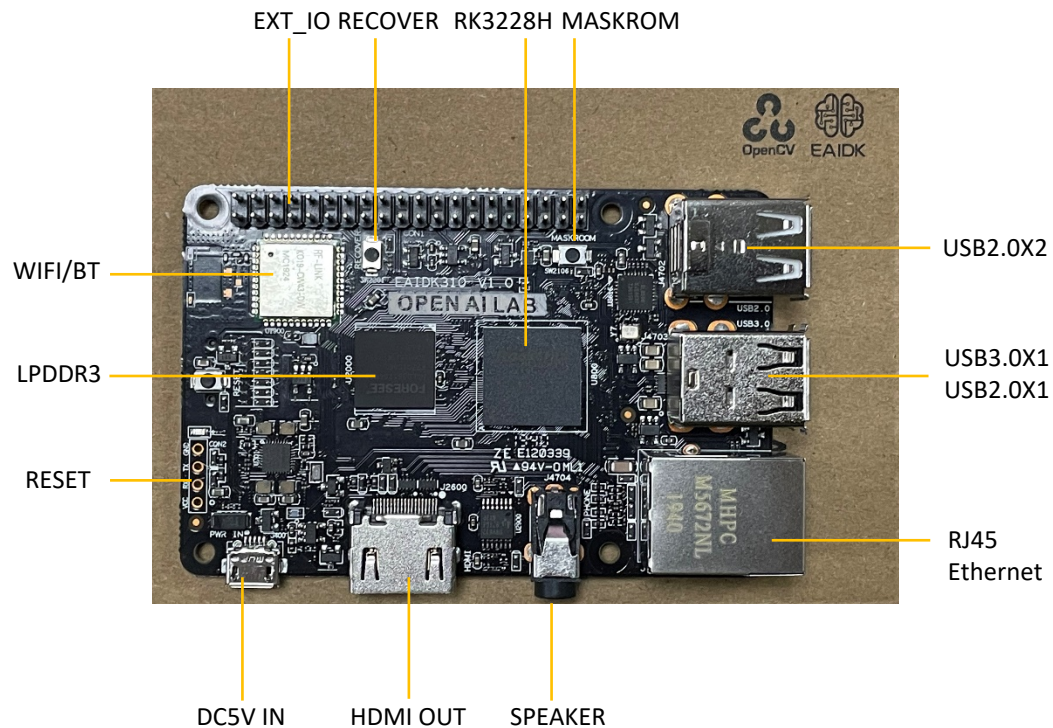
usb camera



## Registration



Face Recognition System Architecture



EAIDK-310

Hardware specification	
SoC	RK3228H
CPU	ARM 4-core Cortex-A53, 64-bit processor, up to 1.3GHz
GPU	ARM Mali-450 MP2 GPU with OpenGL ES 1.1/2.0 and OpenVG 1.1
Running memory	LPDDR3 1 GB
Built-in storage	Standard without eMMC, optional 8GB high speed eMMC
Extended storage	MicroSD, up to 128GB
Wired network	RJ45, 10/100M adaptive
WIFI	802.11 ac/a/b/g/n, 2.4G/5GHz
Bluetooth	Bluetooth 5.0
USB	1xUSB3, 3xUSB2, 1xMicro-USB
HDMI	2.0, 1xType-A, up to 4Kx2K@60Hz
Debug interface	UART (TTL level)
Other interface	2x12C 1xSPI 9xGPIO 2xGPIO (output only) 1xSpeaker
power supply	Micro USB 5V/2A

# OpenCV DNN Module

- Inference framework
- Compatible to many popular deep learning frameworks

Lightness

Convenience



Caffe



- Run on multiple devices: CPU, GPU, VPU, FPGA
- Run on different OS: Linux, Windows, Android, MacOS
- Backend framework to leverage different acceleration libs

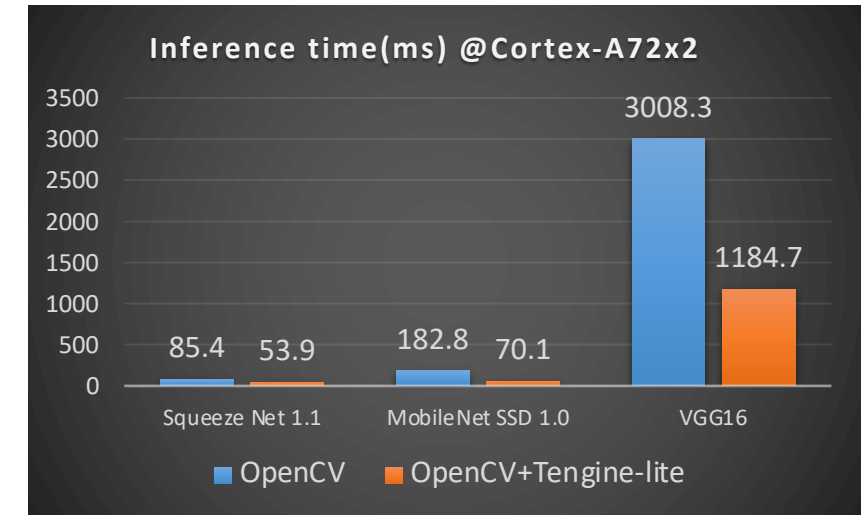
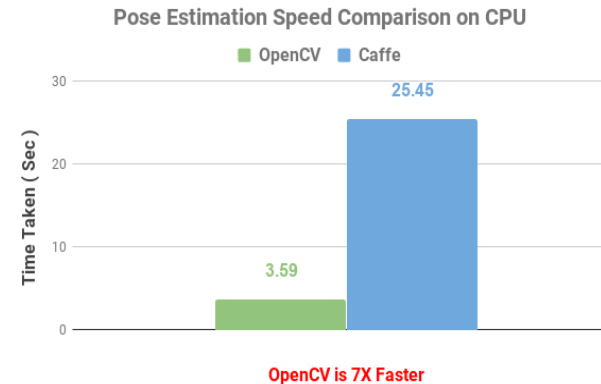
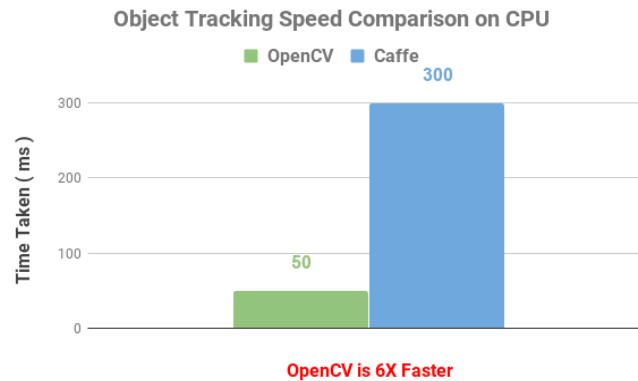
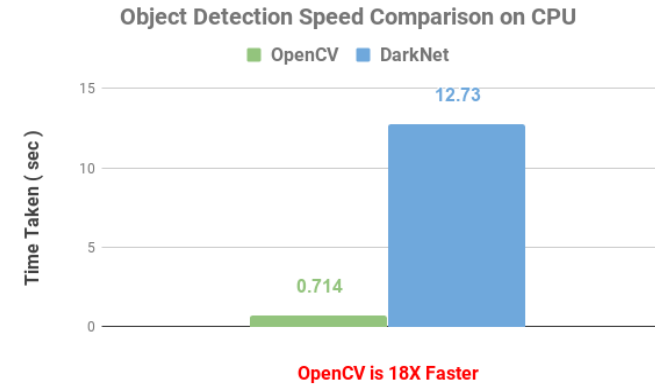
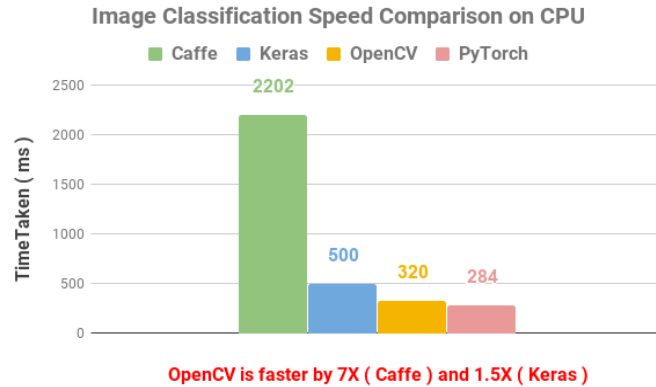
Universality

Accelerated

# well tested networks

Image Classification		Object Detection		Semantic Segmentation	Pose Estimation	Image Processing	Person Identification	Text Detection and Recognition	Depth Estimation
<b>Caffe</b> AlexNet GoogLeNet VGG ResNet SqueezeNet DenseNet ShuffleNet  <b>TensorFlow</b> Inception MobileNet EfficientNet  <b>Darknet</b> darknet19	<b>ONNX</b> AlexNet GoogleNet CaffeNet RCNN_ILSVRC13 ZFNet512 VGG16 VGG16_bn ResNet-18v1 ResNet-50v1 CNN Mnist MobileNetv2 LResNet100E-IR Emotion FERPlus Squeezenet DenseNet121 Inception v1, v2 Shufflenet	<b>Caffe</b> SSD VGG MobileNet-SSD Faster-RCNN R-FCN OpenCV Face Detector  <b>TensorFlow</b> SSD Faster-RCNN Mask-RCNN EAST EfficientDet	<b>Darknet</b> YOLOv2 tiny YOLO YOLOv3 tiny YOLOv3 YOLOv4 tiny YOLOv4  <b>ONNX</b> TinyYolov2 PyTorch YOLOv3 PyTorch YOLOv3-tiny PyTorch SSD VGG	<b>Caffe</b> FCN  <b>Torch</b> ENet  <b>ONNX</b> ResNet101_DUC_HDC  <b>TensorFlow</b> DeepLab  <b>Pytorch</b> UNet DeepLabV3 FPN UNetPlus BiSeNet	<b>Caffe</b> OpenPose  <b>PyTroch</b> AlphaPose	<b>Caffe</b> Colorization  <b>Torch</b> Fast-Neural-Style  <b>ONNX</b> Style Transfer	<b>Torch</b> OpenFace  <b>PyTorch</b> Torchreid  <b>TensorFlow</b> MoibleFaceNet	<b>PyTorch</b> EasyOCR CRNN	<b>PyTorch</b> Monodepth2

# Speed comparison



## Tengine:

- edge AI inference framework.
- Supports CPU, GPU, DSP, NPU.
- Supports TensorFlow, Caffee, MxNet, PyTorch, MegEngine, DarkNet, ONNX, ncnn.

AWS **t2.large** instance (2 vCPUs and 8 GB of RAM, no GPU), Ubuntu 16.04 LTS

<https://www.learnopencv.com/cpu-performance-comparison-of-opencv-and-other-deep-learning-frameworks/>



# Easy-to-use

```

cv::dnn::Net net = cv::dnn::readNet(model_name, model_config); ..... load dl model
net.setPreferableBackend(cv::dnn::DNN_BACKEND_DEFAULT); ..... choose accelerating backend
net.setPreferableTarget(cv::dnn::DNN_TARGET_CPU); ..... set target device

cv::Mat blob = cv::dnn::blobFromImage(img, scale_factor, size, mean, ...);
net.setInput(blob);
net.forward(output_blobs, output_names); ..... inferencing

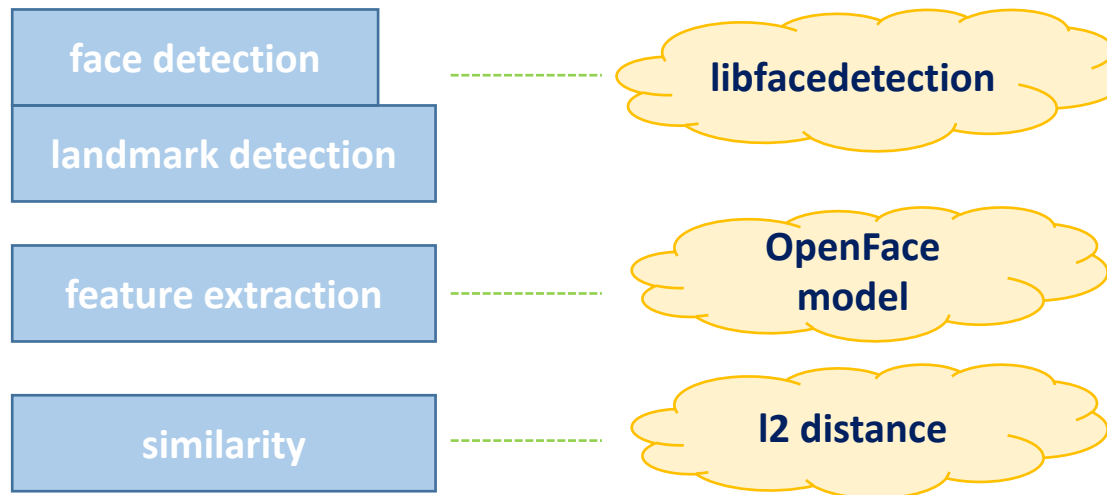
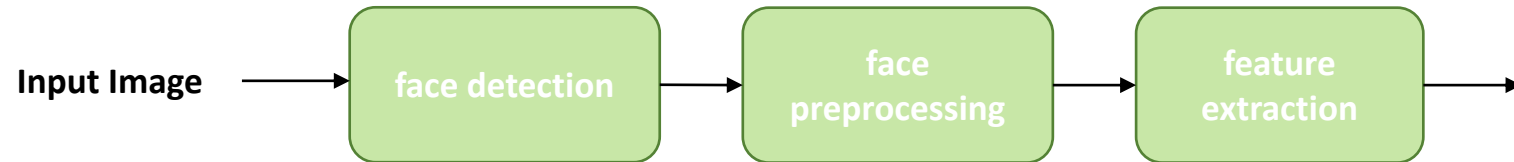
```

DNN_BACKEND_DEFAULT
Python: cv.dnn.DNN_BACKEND_DEFAULT
DNN_BACKEND_HALIDE
Python: cv.dnn.DNN_BACKEND_HALIDE
DNN_BACKEND_INFERENCE_ENGINE
Python: cv.dnn.DNN_BACKEND_INFERENCE_ENGINE
DNN_BACKEND_OPENCV
Python: cv.dnn.DNN_BACKEND_OPENCV
DNN_BACKEND_VKCOM
Python: cv.dnn.DNN_BACKEND_VKCOM
DNN_BACKEND_CUDA
Python: cv.dnn.DNN_BACKEND_CUDA

DNN_TARGET_CPU
Python: cv.dnn.DNN_TARGET_CPU
DNN_TARGET_OPENCL
Python: cv.dnn.DNN_TARGET_OPENCL
DNN_TARGET_OPENCL_FP16
Python: cv.dnn.DNN_TARGET_OPENCL_FP16
DNN_TARGET_MYRIAD
Python: cv.dnn.DNN_TARGET_MYRIAD
DNN_TARGET_VULKAN
Python: cv.dnn.DNN_TARGET_VULKAN
DNN_TARGET_FPGA
Python: cv.dnn.DNN_TARGET_FPGA
DNN_TARGET_CUDA
Python: cv.dnn.DNN_TARGET_CUDA
DNN_TARGET_CUDA_FP16
Python: cv.dnn.DNN_TARGET_CUDA_FP16
DNN_TARGET_HDDL
Python: cv.dnn.DNN_TARGET_HDDL



# Now let's try to build the system



Speed can reach **1000FPS**.

<https://github.com/ShiqiYu/libfacedetection>

# face and landmark detection

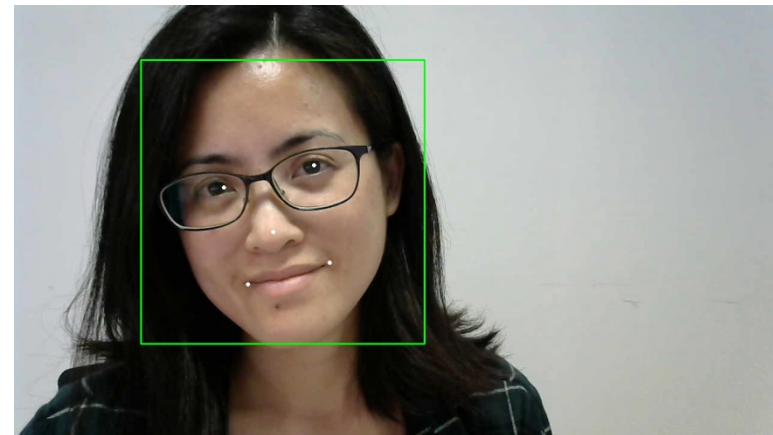
```
// Load .onnx model using OpenCV's DNN module
cv::dnn::Net net = cv::dnn::readNet(argv[2]);
net.setPreferableBackend(cv::dnn::DNN_BACKEND_DEFAULT);
net.setPreferableTarget(cv::dnn::DNN_TARGET_CPU);

// Build blob
blob = cv::dnn::blobFromImage(im, 1.0, cv::Size(128, 96), cv::Scalar());

// Forward
net.setInput(blob);
net.forward(output_blobs, output_names);

// Decode bboxes, landmarks and scores
dets = pb.decode(output_blobs[0], output_blobs[1], output_blobs[2], conf_thresh);

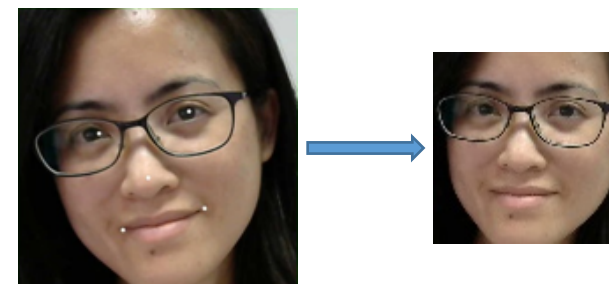
// NMS
if (dets.size() > 1) {
    nms(dets, nms_thresh);
    if (dets.size() > keep_top_k) { dets.erase(dets.begin()+keep_top_k, dets.end()); }
}
else if (dets.size() < 1) {
    std::cout << "No faces found." << std::endl;
}
```





# face alignment

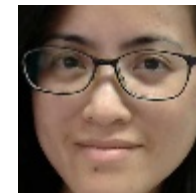
```
void faceAlignment(const Mat& img, Mat& faceImgAligned,  
                  float* eyeCenters, float* eyeCenters_ref,  
                  Size faceSize)  
{  
    float dist_ref = eyeCenters_ref[2] - eyeCenters_ref[0];  
    float dx = eyeCenters[2] - eyeCenters[0];  
    float dy = eyeCenters[3] - eyeCenters[1];  
    float dist = sqrt(dx * dx + dy * dy);  
  
    // scale  
    double scale = dist_ref / dist;  
    // angle  
    double angle = atan2(dy, dx) * 180 / PI;  
    // center  
    Point2f center = Point2f(0.5 * (eyeCenters[0] + eyeCenters[2]),  
                             0.5 * (eyeCenters[1] + eyeCenters[3]));  
    // calculate rotation matrix  
    Mat rot = getRotationMatrix2D(center, angle, scale);  
    // translation  
    rot.at<double>(0, 2) += faceSize.width * 0.5 - center.x;  
    rot.at<double>(1, 2) += eyeCenters_ref[1] - center.y;  
  
    // apply affine transform  
    cv::Mat imgIn = img.clone();  
    imgIn.convertTo(imgIn, CV_32FC3, 1. / 255.);  
    warpAffine(imgIn, faceImgAligned, rot, faceSize);  
    faceImgAligned.convertTo(faceImgAligned, CV_8UC3, 255);  
}
```



# feature extraction

```
// load feature extractor model
Net featEmbedder = readNet(model_dir + "openface/openface_nn4.small2.v1.t7");
featEmbedder.setPreferableBackend(DNN_BACKEND_OPENCV);
featEmbedder.setPreferableTarget(DNN_TARGET_CPU);

// feature extraction
Mat blob_faceAligned;
blobFromImage(faceAligned, blob_faceAligned, 1. / 255., Size(), Scalar(), true, false);
featEmbedder.setInput(blob_faceAligned);
Mat featA = featEmbedder.forward();
```



# calculate similarity

```
cv::norm(featA - featB)
```



# Outline

- Face recognition in brief
- Build a FR system on ARM board using OpenCV
- **Assignment**



# Assignment

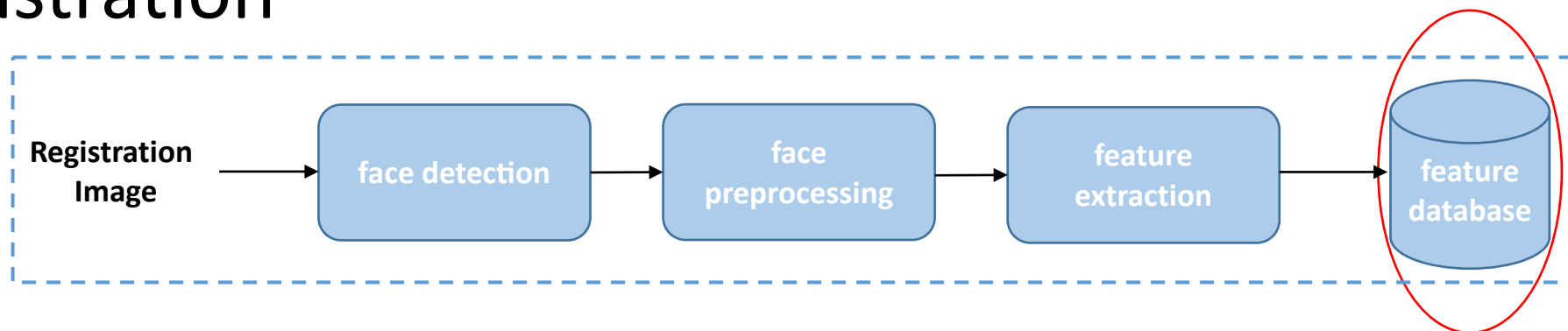
- A. Build a complete face recognition system on ARM board using OpenCV. Write a report in English about the system.
- B. Build any other kind of biometric recognition system on ARM board using OpenCV. Write a report in English about the system.

- Submit to: [jia.wu@opencv.org.cn](mailto:jia.wu@opencv.org.cn)
- Deadline: 23:59 Jan. 27, 2021
- Top 3 teams will be awarded the advanced EAIDK-610 Kit.



# Issues

- Training
  - use deep learning frameworks
- Registration



- UI

# Thank You!



[www.opencv.org.cn](http://www.opencv.org.cn)