Original Research Article

Deployment and Implementation of Facial Feature Recognition Algorithm Based on RP-rv1126 Development Board

Chao Li [1], Linlu Zou[2]

1 Cethik Group Co.,Ltd, Hangzhou, Zhejiang, 311121, China

2 City college, Zhejiang University, Hangzhou, Zhejiang, 310015, China

Abstract: Firstly, it is necessary to use the Rockchip Micro RKNN tool to convert the inference model into an RKNN model that can run on edge devices. At the same time, a C++main program should be written to facilitate calling the RKNN model for feature inference on specific images; Configure the compilation options for the development board, enable the NPU mini driver, and after successful image compilation, generate an image file with the NPU driver enabled and re burn the development board. Finally, by uploading the main program, rknn inference model, and the image to be inferred to the development board, the inference of the algorithm on the development board can be achieved. Through actual testing, its model inference time is approximately 0.18s, which can meet the needs of facial feature recognition in certain scenarios.

Keyword: Facial Feature Recognition; RP-rv1126; RKNN; NPU; Algorithm deployment

1. Introduction

Human facial feature recognition is an important research direction in the field of artificial intelligence. Generally speaking, it can be achieved through large-scale data collection and annotation, followed by model training and generation based on supervised machine learning methods. Then, the generated algorithm model can be used to perform algorithmic inference on the images captured by the camera to obtain facial feature recognition results. Facial feature recognition can be used for driver fatigue detection and user emotion detection of home service robots, thereby providing emotional intervention and comfort to humans.

Facial feature recognition of emotions is an important way of emotion calculation. In addition, it can also recognize human emotions through body movements, speech intonation, EEG, and other methods.

The technical route of emotion calculation can roughly cover the main links under its jurisdiction

Data collection: Use a camera to capture facial expressions and eye movements, a microphone to capture voice information, a radar chip to capture breathing and heartbeat information, an EEG cap to collect EEG signals, and an ECG device to collect ECG signals.

Feature extraction:

The features extracted from facial expressions mainly include texture features, statistical features, transformation features, and Harr matrices;

Speech features include prosodic features, sound quality features, and spectral features.

Respiratory and heartbeat characteristics include respiratory and heartbeat frequency, respiratory and heartbeat frequency, rate, and amplitude.

Electrocardiogram signals mainly include time-domain features, frequency-domain features, time-frequency domain features, spatial domain features, etc.

Machine learning, mainly including shallow machine learning and deep machine learning

Shallow machine learning includes support vector machine (SVM), K-nearest neighbor algorithm (KNN), decision tree (DT)

Naive Bayes (NB) and Linear Discriminant Analysis (LDA), among others.

Deep machine learning mainly includes artificial neural networks (ANN), recurrent neural networks (RNN), and convolutional neural networks (CNN)

Hidden Markov Models, Long Short Term Memory Networks (LSTM), etc.

Convert into classification problem (discrete emotion model) and regression problem (continuous emotion model)

Classification problem is a task of predicting discrete labels or categories. In classification problems, the goal of the model is to map the input to predefined categories or labels.

Regression problem is a task of predicting continuous numerical outputs. In this type of problem, the goal of the model is to predict a continuous value based on input features.

In summary, the most important method in facial expression recognition is through facial features. By collecting facial data, extracting features, and using traditional or deep learning machine learning methods, facial expression recognition can be transformed into a discrete classification problem or a continuous regression problem. In this paper, we use a discrete model and deep learning methods to transform facial expression recognition into a classification problem, and deploy the model on edge devices.

2. Related work

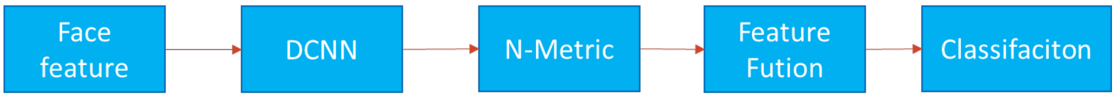
Reference 1[1]Based on the Atlas 200DK embedded device, a lightweight facial expression recognition model Light FER was designed on the basis of the Mobile Vi T framework. This model has low parameter count, low computational complexity, and high performance, which is conducive to deployment on embedded devices. An embedded facial expression recognition system has been implemented in the classroom environment, which can be used to assist teachers in evaluating the quality of classroom teaching. The offline deployment of the model on embedded devices not only solves the network communication limitations brought by cloud deployment, but also addresses user privacy and security issues. By adopting a streamlined algorithm model, it can be deployed and run on embedded devices.

Reference 2[2] mentioned that due to the long physical distance between the data acquisition end and the data processing end in cloud computing solutions, there is a huge bandwidth pressure and difficulty in achieving real-time performance. However, due to the limited hardware resources of edge devices, deep learning applications are difficult to run efficiently on edge devices without optimization. Its high utilization of computing and storage resources is also not conducive to the realization of real-time characteristics. Currently, deep learning inference tasks are implemented from three directions using limited hardware resources: lightweight model structure design, model compression technology, and neural networks. High performance computing for inference. Long short-term memory networks were optimized using tensor compression techniques. This greatly reduces the overall size of the long short-term memory network model, with a very high compression ratio, thereby alleviating storage space size; On the other hand, the continuous computation of small-sized core tensors is greatly reduced. The parameter size of the classifier based on long short-term memory network has been reduced from millions to thousands, and the storage space occupied has been compressed from over 600 megabytes to about 3 megabytes. Its frame rate per second on NVIDIA 1080Ti GPU reaches 45.5, which is about 59.6% faster than the implementation without tensor compression.

Reference 3[3],In the field of Electronic and Communication Engineering (Professional Degree), this paper proposes algorithm improvements on the existing NPD (Normalized Pixel Difference) face detection technology by adding face detection window restrictions and a detection scoring system, resulting in higher efficiency and faster detection speed of the detection algorithm, making it more suitable for low performance ARM terminal processing devices. A new lightweight network model structure, MobileNet, is used to train an expression recognition model. The system collects expression images through a Raspberry Pi camera and sends them to an ARM based face detection module for face detection.

3. The content of this article

3.1. Expression recognition algorithm steps

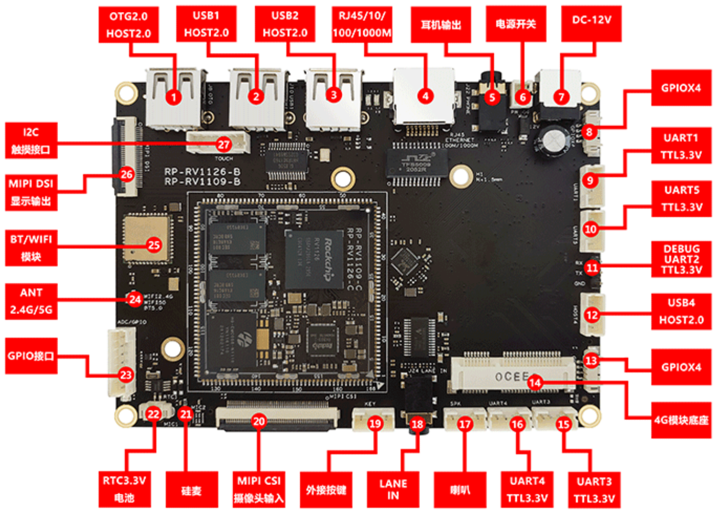


a) Extract CNN features of the face using the VGG16 Face model;

b) In the DCNN block and N-Metric block, different branches are formed based on different threshold τ sequences and fixed FC layers, and a triplet loss function is used to learn an embedding;

c) Finally, the features learned from different branches in ② are fused and image classification is completed using the Softmax logistic regression function.

3.2. Edge Hardware Device-RP-rv1126 Development Board



RV1126 adopts Ruixin Micro RV1126 Cortex-A7 quad core processor+2.0TOPS computing power NPU, equipped with Linux+QT system, with a main frequency of 1500MHz and low heat generation. The core board adopts stamp hole connection method for more reliable welding, supports multiple peripheral extensions, and can be used for facial recognition and AI artificial intelligence image processing.

At the same time, Rockchip Micro provides users with a development kit RKNN Toolkit for model conversion, inference. Users can easily complete the following functions through the Python interface provided by this tool:

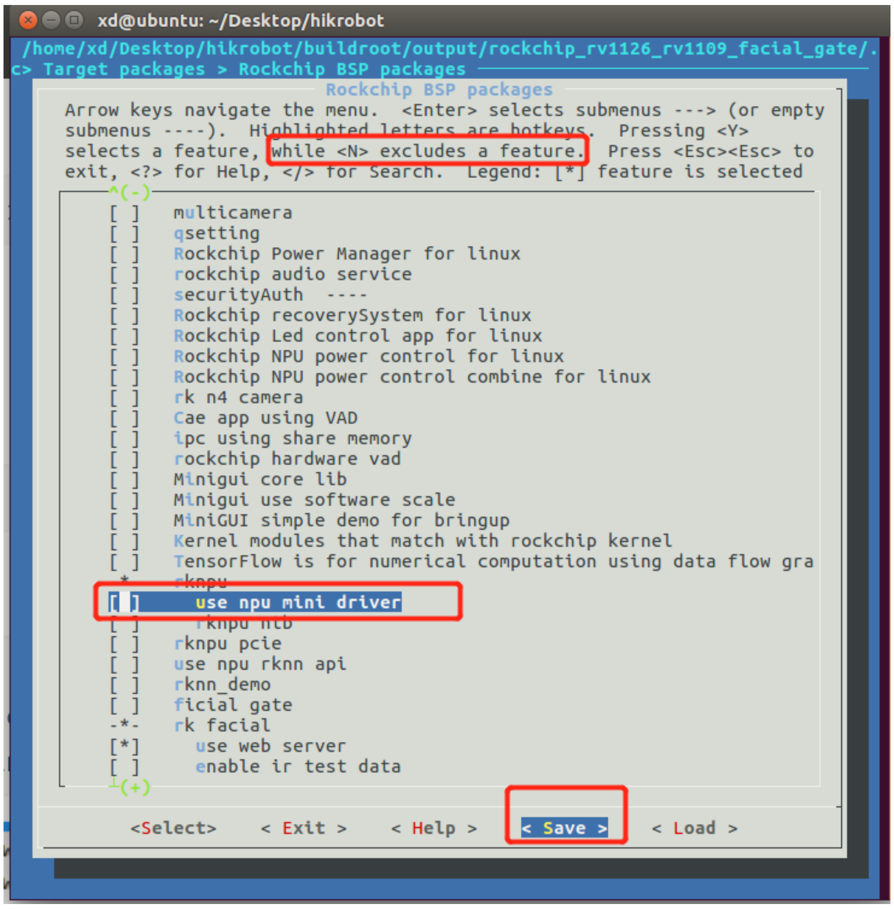
Model conversion: Supports the conversion of models such as Caffe, TensorFlow, TensorFlowLite, ONNX, DarkNet, PyTorch to RKNN models, and supports the import and export of RKNN models. RKNN models can be loaded and used on the RockchipNPU platform.

Model inference: able to simulate RockchipNPU running RKNN model on PC and obtain inference results.

3.3. Edge deployment process of algorithms

Firstly, configure the development environment and install TensorFlow and rknn tools, as well as the cross compilation toolchain, in the Ubuntu environment for desktop development. It should be noted that the NPU of the buildroot environment that comes with this development hardware platform is configured as mini by default and does not come with rknn_derver, so it cannot be simulated by connecting to the motherboard through the PC end

Need to enable NPU compilation option:



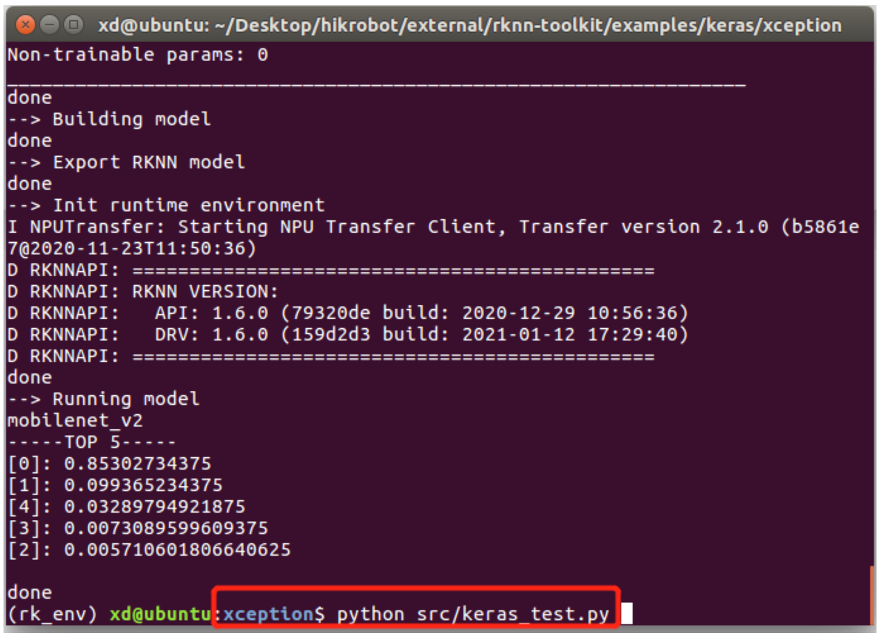
```
xd@ubuntu: ~/Desktop/hikrobot
/home/xd/Desktop/hikrobot/buildroot/output/rockchip_rv1126_rv1109_facial_gate/.
c> Target packages > Rockchip BSP packages
                          Rockchip BSP packages
   Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty
   submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
   selects a feature, while <N> excludes a feature.  Press <Esc><Esc> to
   exit, <?> for Help, </> for Search.  Legend: [*] feature is selected
    ^(-)
       [ ]   multicamera
       [ ]   qsetting
       [ ]   Rockchip Power Manager for linux
       [ ]   rockchip audio service
       [ ]   securityAuth  ----
       [ ]   Rockchip recoverySystem for linux
       [ ]   Rockchip Led control app for linux
       [ ]   Rockchip NPU power control for linux
       [ ]   Rockchip NPU power control combine for linux
       [ ]   rk n4 camera
       [ ]   Cae app using VAD
       [ ]   ipc using share memory
       [ ]   rockchip hardware vad
       [ ]   Minigui core lib
       [ ]   Minigui use software scale
       [ ]   MiniGUI simple demo for bringup
       [ ]   Kernel modules that match with rockchip kernel
       [ ]   TensorFlow is for numerical computation using data flow gra
        *    rknpu
       [ ]     use npu mini driver
       [ ]     rknpu ntb
       [ ]   rknpu pcie
       [ ]   use npu rknn api
       [ ]   rknn_demo
       [ ]   ficial gate
       -*-   rk facial
       [*]     use web server
       [ ]     enable ir test data
    +(+)

        <Select>    < Exit >    < Help >    < Save >    < Load >
```

Cancel the selection of NPU mini driver, save the configuration, and recompile.

After compiling buildroot, rootfs will be automatically compiled to generate the image file rootfs.img. After generating the image file, according to the development documentation, uboot (generate uboot. img) and kenal (generate boot. img) will be compiled separately, and then buildboot will be recompiled. Note that the mini option needs to be removed during compilation. After the normal compilation of Buildboot, if everything is normal. The image file will be generated in the rockdev/directory, and after re burning, the device has already installed the completed NPU driver. At this point, online compilation and simulation can be performed.

Now that the complete version of NPU has been installed, try to perform online compilation, modify the running parameters, and perform joint debugging between the PC and development board. The running command is as follows:



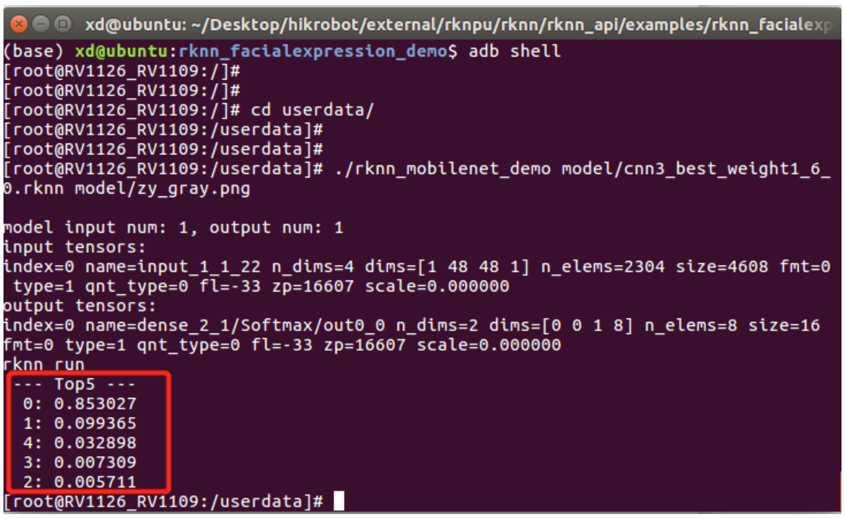
```
xd@ubuntu: ~/Desktop/hikrobot/external/rknn-toolkit/examples/keras/xception
Non-trainable params: 0
_________________________________________________________________
done
--> Building model
done
--> Export RKNN model
done
--> Init runtime environment
I NPUTransfer: Starting NPU Transfer Client, Transfer version 2.1.0 (b5861e
7@2020-11-23T11:50:36)
D RKNNAPI: ==============================================
D RKNNAPI: RKNN VERSION:
D RKNNAPI:   API: 1.6.0 (79320de build: 2020-12-29 10:56:36)
D RKNNAPI:   DRV: 1.6.0 (159d2d3 build: 2021-01-12 17:29:40)
D RKNNAPI: ==============================================
done
--> Running model
mobilenet_v2
-----TOP 5-----
[0]: 0.85302734375
[1]: 0.099365234375
[4]: 0.03289794921875
[3]: 0.0073089599609375
[2]: 0.005710601806640625

done
(rk_env) xd@ubuntu:xception$ python src/keras_test.py
```

Normal results indicate that the NPU firmware update has taken effect;

4. Experiment

The facial expression recognition algorithm is deployed on edge devices. After testing, the facial expression recognition algorithm executed by edge devices shows good inference results when emotions are divided into positive and negative emotions, achieving a recognition rate of $\geq$ 90%. However, when emotions are further subdivided into anger, sadness, and other emotions, the inference effect of the algorithm is not ideal, indicating that the data model has little differentiation between these two emotions. Further expanding the dataset may improve the recognition effect.



```
xd@ubuntu: ~/Desktop/hikrobot/external/rknpu/rknn/rknn_api/examples/rknn_facialexp
(base) xd@ubuntu:rknn_facialexpression_demo$ adb shell
[root@RV1126_RV1109:/]#
[root@RV1126_RV1109:/]#
[root@RV1126_RV1109:/]# cd userdata/
[root@RV1126_RV1109:/userdata]#
[root@RV1126_RV1109:/userdata]#
[root@RV1126_RV1109:/userdata]# ./rknn_mobilenet_demo model/cnn3_best_weight1_6_
0.rknn model/zy_gray.png

model input num: 1, output num: 1
input tensors:
index=0 name=input_1_1_22 n_dims=4 dims=[1 48 48 1] n_elems=2304 size=4608 fmt=0
 type=1 qnt_type=0 fl=-33 zp=16607 scale=0.000000
output tensors:
index=0 name=dense_2_1/Softmax/out0_0 n_dims=2 dims=[0 0 1 8] n_elems=8 size=16
fmt=0 type=1 qnt_type=0 fl=-33 zp=16607 scale=0.000000
rknn run
 --- Top5 ---
  0: 0.853027
  1: 0.099365
  4: 0.032898
  3: 0.007309
  2: 0.005711
[root@RV1126_RV1109:/userdata]#
```

At the same time, the pure inference time after printing the model load in the program is approximately 0.18s, which means that the inference efficiency of this model on the edge device rp-rv1126 is approximately 5fps.

References

[1] Master's Electronic Journal Publication Information: Year: Issue 04, 2024, University of Electronic Science and Technology of China, Zhong Haoyuan, Research and Application of Lightweight Facial Expression Recognition Model for Embedded Devices

[2] Implementation of Video Expression Recognition Based on Deep Compressed Spatiotemporal Model on Edge Devices, Master's Electronic Journal Publication Information: Year: Issue 01, 2021

[3] Design and Implementation of ARM based Facial Expression Recognition System, Issue 02, 2020, Electronic