M480 Keyword spotting

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	本范例代码使用机器学习的方式在M480上实现关键词辨识(Keyword spotting)功能
BSP 版本	M480 Series BSP CMSIS V3.04.000
开发平台	NuMaker-PFM-M487 Ver 3.0

1 功能介绍

1.1 简介

一套完整的深度学习语音识别系统使用两个平台完成如图1-1，一为PC端，利用TensorFlow与Python撰写完整的深度学习程序代码并训练模型，因本方案使用的学习模式为监督式学习，因此需给系统大量的训练数据和卷标(Labels)，接着将撷取到的特征用深度神经网络(Deep Neural Networks, DNN)模型来训练，并反复修正训练的模型，直到模型达到系统优化的状态；二为NuMaker-PFM-M487平台，使用NuMicro® M480系列芯片下的NuMaker-PFM-M487开发版，利用PC建出来的深度学习模型与训练结果(特征参数)，应用到NuMaker-PFM-M487平台来完成可实时的语音识别系统。

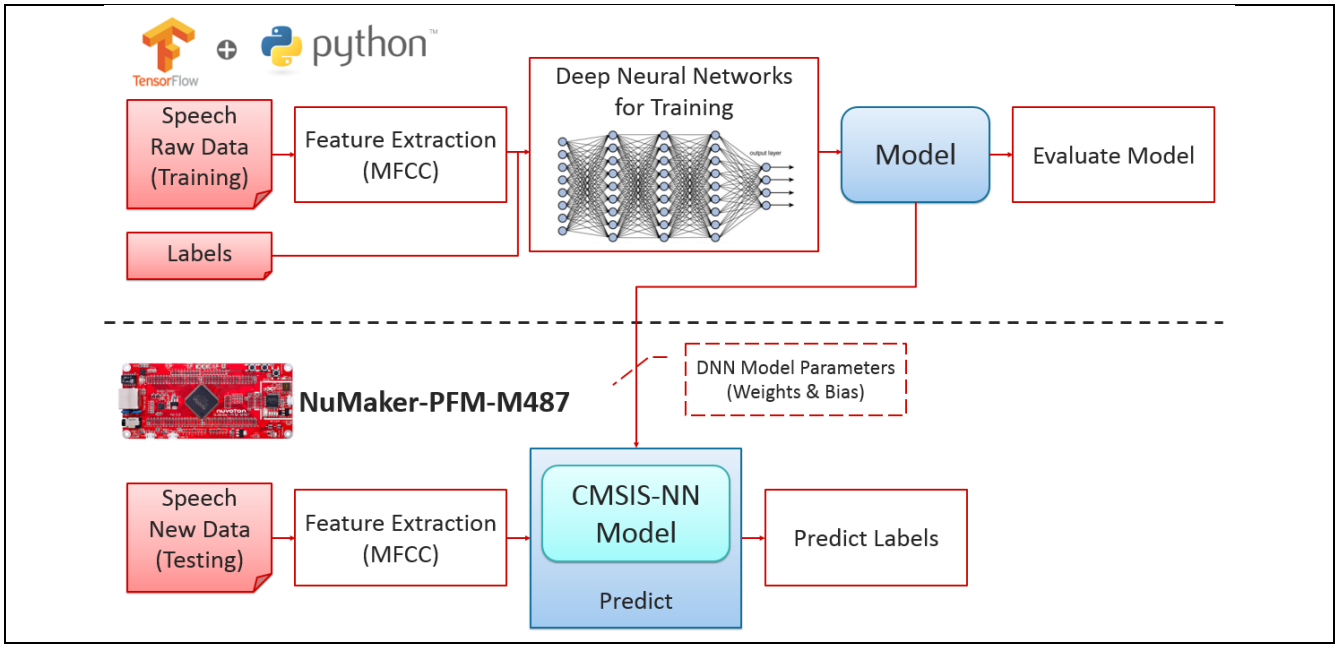


图 1-1 语音识别系统流程图

更详细说明，请参考应用手册：

AN_0029_M480series_SpeechRecognition_MachineLearning_Rev1.00

1.2 深度学习介绍

机器学习(Machine Learning)是人工智能(Artificial Intelligence, AI)的分支，运作流程是透过算法，使用大量数据进行训练，训练完成后会产生模型，当未来有新的数据时，我们可以使用训练产生的模型进行预测。机器学习应用相当广泛，例如：推荐引擎、定向广告、需求预测、垃圾邮件过滤、医学诊断、自然语言处理、搜索引擎、诈骗侦测、证券分析、视觉辨识、语音识别、手写识别等等。

而深度学习(Deep Learning)又是机器学习的分支，是人工智能中，成长最快的领域，至今已有数种深度学习的框架，如深度神经网络(Deep Neural Networks, DNN)、卷积神经网络(Convolutional Neural Networks, CNN)、递归神经网络(Recurrent Neural Networks, RNN)等等，何时该用什么样的架构，答案是没有一定的，需要依据数据的大小、性质、可接受的计算时间、学习的紧迫性、想用这些数据做什么来判断。深度学习特别在视觉辨识、语音识别、自然语言处理、生物医学等领域的应用上，有非常好的效果。

深度学习就是让机器模拟人脑的运作方式，进而和人类一样具备学习的能力，不过由于人类神经网络太过复杂，所以为了方便以机器模拟，将神经元分为多层次，来仿真神经网络，神经网络通常会有1个输入层(Input Layer)、1个输出层(Output Layer)和多个可以被训练的隐藏层(Hidden Layer)，超过2个隐藏层就可以被称为深度学习。

1.2.1 深度神经网络(Deep Neural Networks, DNN)介绍

深度神经网络(Deep Neural Networks, DNN)是深度学习最基本的一种判别模型，可以使用反向传播(Backpropagation)算法进行训练，权重(Weight)更新可以使用下式进行梯度下降法(Gradient descent)迭代式进行更新：

$$\Delta w_{ij}(t+1) = \Delta w_{ij}(t) + \eta \frac{\partial C}{\partial w_{ij}}$$

其中，η为学习率(Learning Rate)，C为损失函数(Loss Function)，函式的选择与活化函数(Activation Function)相关，以本文件举例，为了在一个多分类(Multi Class Classification)问题上进行监督式学习，选择使用线性整流函数(Rectified Linear Unit, ReLU)作为活化函数，使用交叉熵(Cross Entropy)作为损失函数，交叉熵的定义如下：

$$C = -\sum_{j} d_j \log(p_j)$$

其中d_j代表输出单元j的目标机率，p_j代表应用活化函数后对单元j的机率输出；而输出层的归一化指数函数(Softmax Function)定义如下：

$$p_j = \frac{exp(x_i)}{\sum_k exp(x_k)}$$

其中p_j代表类别j的机率，而x_i和x_k分别是对单元i和k的输入。

深度神经网络的网络架构如图1-2。

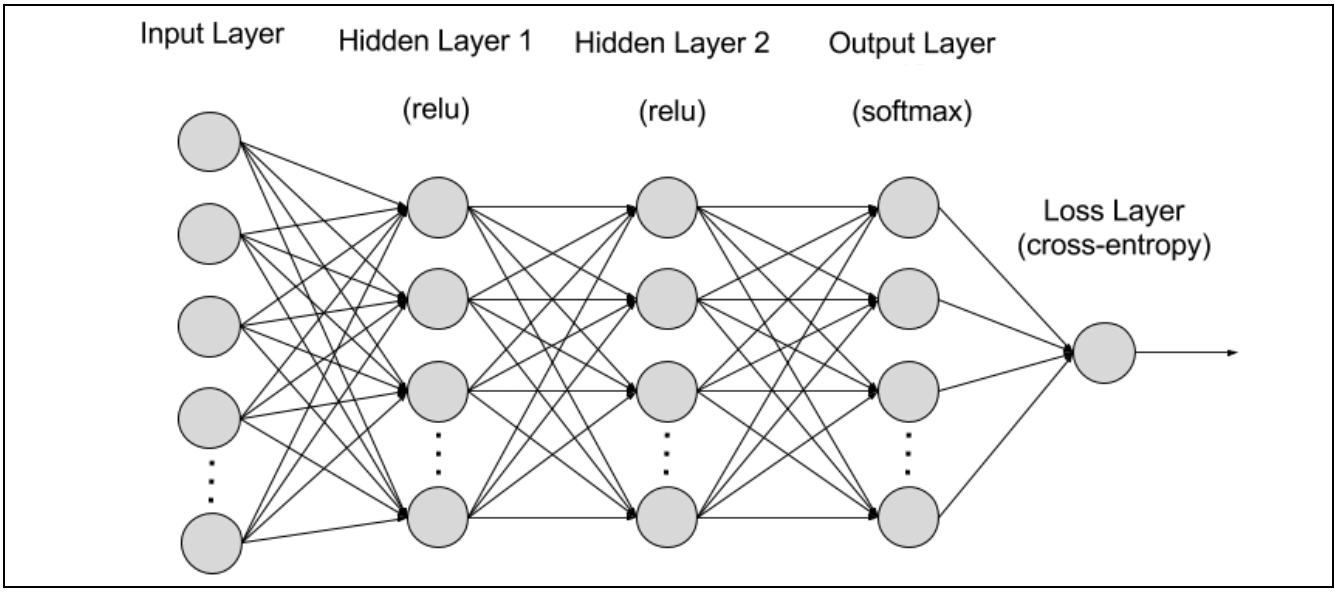


图 1-2 深度学习网络(DNN)架构图

1.3 TensorFlow 简介

TensorFlow是由Google提供的开放源代码链接库，Google有很多产品使用TensorFlow技术开发深度学习与机器学习功能，例如：Gmail过滤垃圾信、Google语音识别、Google 图片辨识、Google翻译…等等，前一小节深度学习的介绍，我们了解深度学习的核心，以张量(矩阵)运算仿真神经网络，因此TensorFlow的主要设计就是让矩阵运算达到最高效能，并且能够在不同的平台上开发。

1.3.1 TensorFlow 架构图

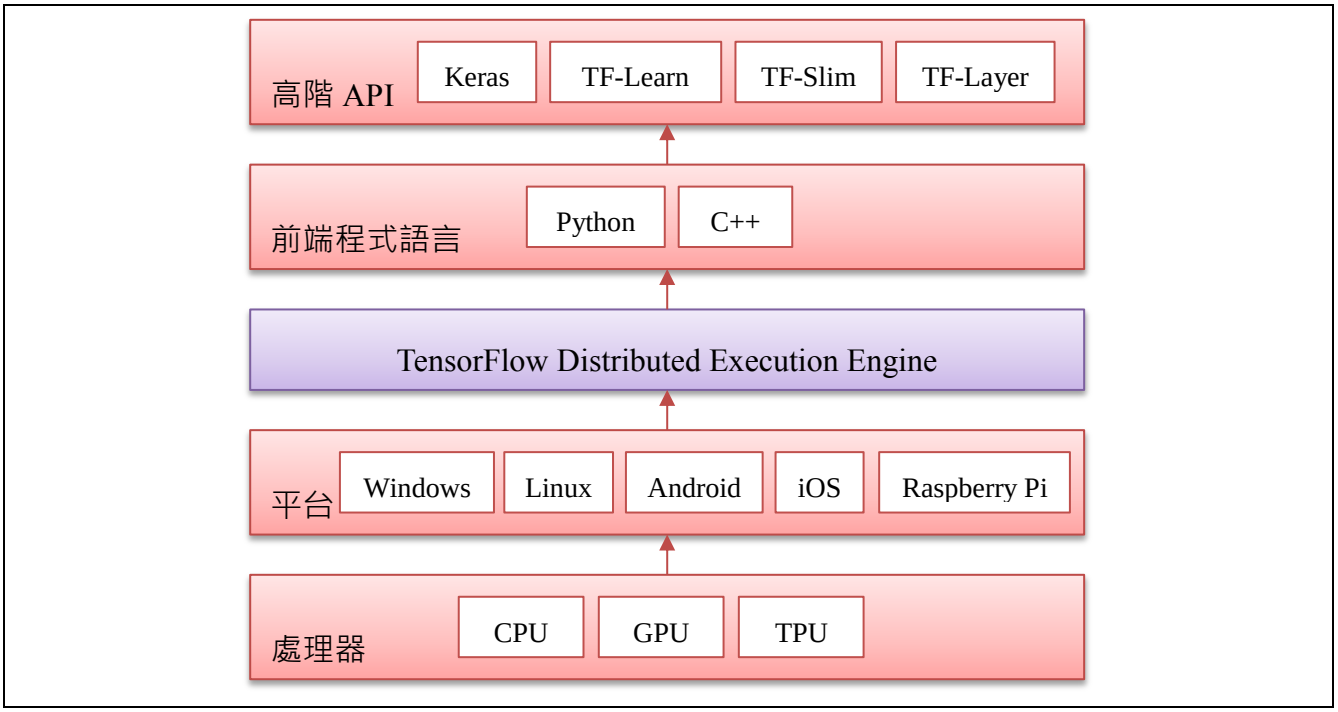


图 1-3 TensorFlow 架构图

以上架构图，由下而上说明如下：

- 处理器：TensorFlow可以在CPU、GPU、TPU上执行
  - CPU：每一台计算机都有中央处理器(CPU)，可以执行TensorFlow，本语音识别的应用使用CPU即可。
  - GPU：图形处理器，有数千个小型且高效率的核心，可以发挥平行运算的强大功能。
  - TPU：Tensor Processing Unit是google為人工智能(AI)研发专属的芯片，比GPU有更好的执行功能。
- 平台：

TensorFlow具跨平台能力，可在目前主流的平台执行，本文件使用Windows 10操作系统。

- TensorFlow Distributed Execution Engine(分布式执行引擎)：

在深度学习中，最耗时间就是模型的训练，尤其大型的深度学习模型，必须使用大量数据进行训练，TensorFlow具备分布式计算能力，可同时在数百台的机器上执行模型训练，大幅缩短模型训练的时间。

- 前端程序语言：

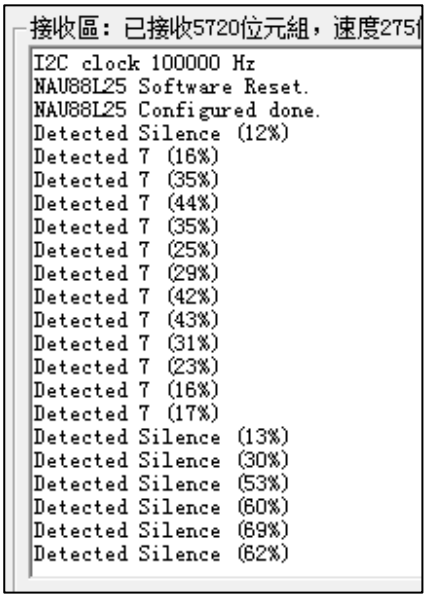
TensorFlow可使用多种语言，而Python具有程序代码简明、易学习、高生产力的特质、面向对象、函数式的动态语言，应用非常广，本文件的深度学习语言也是使用Python来开发。

- 高阶API：

TensorFlow是比较低阶的深度学习API，所以程序设计模型时，必须自行设计张量乘积、卷积等底层操作，因此本文件搭配高阶的API—Keras，让开发者使用更简洁更可读性的程序代码，就可以建构出各种复杂的深度学习模型。

1.4 执行结果

接上麦克风并说出英文数字及会打印出辨识之结果。



2 代码介绍

下图为NuMicro® M480系列微控制器的程序流程图，图中方块为程序中主要的程序，用户可以由图简易地了解程序的架构。

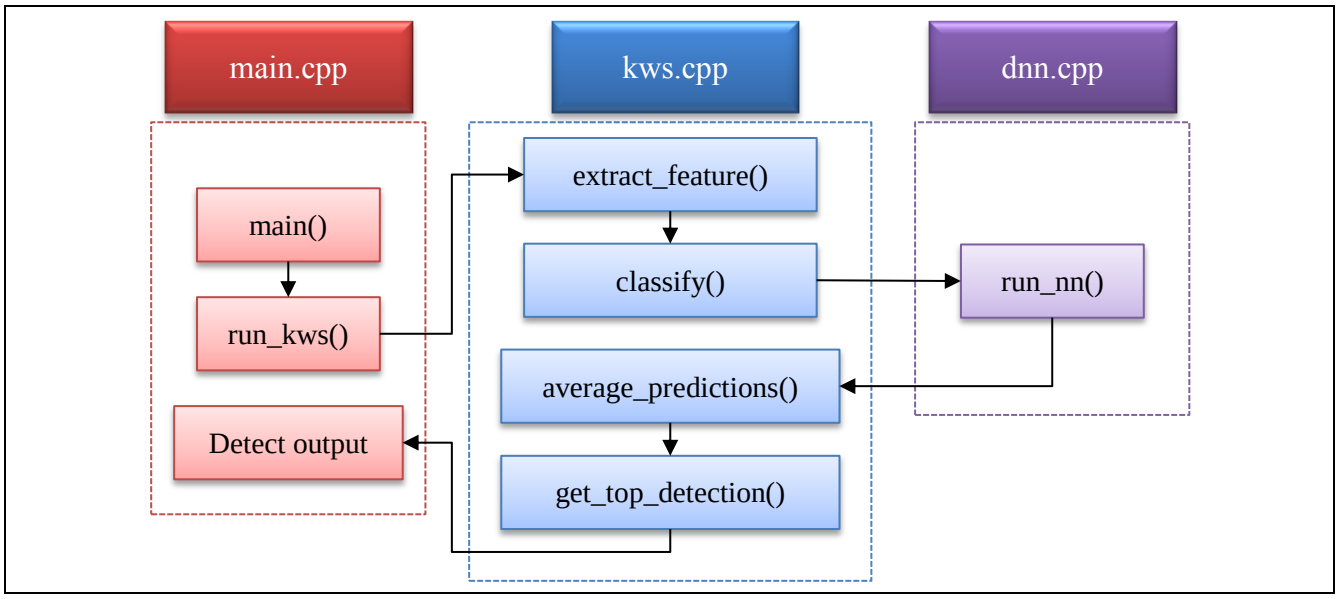


图 2-1 NuMicro® M480 微控制器程序流程图

2.1.1 KWS 范例主程序

```
/* Output Class */
char output_class[12][10] = { "Silence", "Unknown", "One", "Two", "Three", "Four",
"Five" ,"Six", "Seven", "Eight", "Nine", "Zero"};
void run_kws()
{
  //Averaging window for smoothing out the output predictions
  int averaging_window_len = 3;  //i.e. average over 3 inferences or 240ms

  kws->extract_features(2); //extract mfcc features
  kws->classify();          //classify using dnn
  kws->average_predictions(averaging_window_len);
  max_ind = kws->get_top_detection(kws->averaged_output);

  printf("Detected %s\r\n",output_class[max_ind]);
}
```

2.1.2 KWS Function (使用 MFCC 特征撷取和 DNN 模型)

```
/* This overloaded function is used in streaming audio case */
void KWS::extract_features(uint16_t num_frames)
```

```
{
  //move old features left
  memmove(mfcc_buffer,mfcc_buffer+(num_frames*NUM_MFCC_COEFFS),(NUM_FRAMES-
num_frames)*NUM_MFCC_COEFFS);
  //compute features only for the newly recorded audio
  int32_t mfcc_buffer_head = (NUM_FRAMES-num_frames)*NUM_MFCC_COEFFS;
  for (uint16_t f = 0; f < num_frames; f++) {
    mfcc->mfcc_compute(audio_buffer+(f*FRAME_SHIFT),2,&mfcc_buffer[mfcc_buffer_head]);
    mfcc_buffer_head += NUM_MFCC_COEFFS;
  }
}
void KWS::classify()
{
  nn->run_nn(mfcc_buffer, output);

  // Softmax
  arm_softmax_q7(output,OUT_DIM,output);

  //do any post processing here
}

void KWS::average_predictions(int window_len)
{
  //shift right old predictions
  for(int i=window_len-1;i>0;i--) {
    for(int j=0;j<OUT_DIM;j++)
      predictions[i][j]=predictions[i-1][j];
  }
  //add new predictions
  for(int j=0;j<OUT_DIM;j++)
    predictions[0][j]=output[j];
  //compute averages
  int sum;
  for(int j=0;j<OUT_DIM;j++) {
    sum=0;
    for(int i=0;i<window_len;i++)
      sum += predictions[i][j];
    averaged_output[j] = (q7_t)(sum/window_len);
  }
}

int KWS::get_top_detection(q7_t* prediction)
```

```
{
  int max_ind=0;
  int max_val=-128;
  for(int i=0;i<OUT_DIM;i++) {
    if(max_val<prediction[i]) {
      max_val = prediction[i];
      max_ind = i;
    }
  }
  return max_ind;
}
```

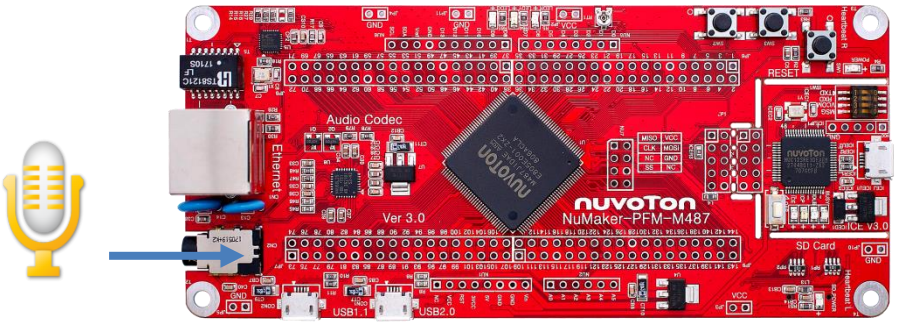
2.1.3 NN Function

```
void DNN::run_nn(q7_t* in_data, q7_t* out_data)
{
// Run all layers
// IP1
arm_fully_connected_q7(in_data, ip1_wt, IN_DIM, IP1_OUT_DIM, 1, 7, ip1_bias, ip1_out,
vec_buffer);
// RELU1
arm_relu_q7(ip1_out, IP1_OUT_DIM);
// IP2
arm_fully_connected_q7(ip1_out, ip2_wt, IP1_OUT_DIM, IP2_OUT_DIM, 2, 8, ip2_bias, ip2_out,
vec_buffer);
// RELU2
arm_relu_q7(ip2_out, IP2_OUT_DIM);
// IP3
arm_fully_connected_q7(ip2_out, ip3_wt, IP2_OUT_DIM, IP3_OUT_DIM, 2, 9, ip3_bias, ip3_out,
vec_buffer);
// RELU3
arm_relu_q7(ip3_out, IP3_OUT_DIM);
// IP4
arm_fully_connected_q7(ip3_out, ip4_wt, IP3_OUT_DIM, OUT_DIM, 0, 6, ip4_bias, out_data,
vec_buffer);
}
```

3 软件与硬件环境

- 软件环境
  - BSP 版本
    - M480 Series BSP CMSIS V3.04.000
  - IDE 版本
    - Keil uVersion 5.26
- 硬件环境
  - 电路组件
    - NuMaker-PFM-M487 or other M480 Development Board
    - 3.5 mm 接头耳机
  - 示意图

在 NuMaker-PFM-M487 的 CN2 插入麦克风。



4 目录信息

EC_M480_KWS_V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
ML_PCTool	Machine Learning python source files
SampleCode	
ExampleCode	Source file of example code

5 如何执行范例程序

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 M480_KWS.uvproj
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
   c. 进入 / 离开除错模式
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
Oct. 23, 2019	1.00	1. 初始发布.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.