ISEE 算法模块接口定义

version 0.3

1. 基本定义

```
/** @file basic_define.h
 *  @brief   Basic definitions used in the code for LPRC.
 *  @author  CRIPAC
 *  @version 0.1
 *  @date    2018/04/08
 */

#ifndef _LCPR_BASIC_DEFINE_H_
#define _LCPR_BASIC_DEFINE_H_

/*Error list.*/
enum LPRC_ErrorsList {
    LCPR_SUCCESS = 0,
    LCPR_LOAD_IMG_FAILED = -1,
    LCPR_LOAD_MODEL_FAILED = -2,
    LCPR_LOAD_WEIGHTS_FAILED = -3,
    LCPR_LOAD_PARAMS_FILE_FAILED = -4,
    LCPR_BAD_VAL = -5,
};

/* Information of input image.*/
typedef struct lcpr_input_image_t {
    int img_w;
    int img_h;
    int num_channels;
    unsigned char* img_data;  // bgr, bgr, bgr, ...
} LCPRInputIMG;

/* Bounding box */
typedef struct lcpr_bounding_box_t {
    int top;
    int left;
    int bottom;
    int right;
} LCPRBoundingBox;

#endif  //_LCPR_BASIC_DEFINE_H_
```

2. 各模块接口定义

(1). 目标检测

```
/** @file pedestrian_detection.h
 *  @brief   Interfaces for pedestrian detection for a single image.
```

```
3.    *  @author  CRIPAC
4.    *  @version 0.1
5.    *  @date    2014/04/09
6.    */
7.   #ifndef _LCPR_PEDESTRIAN_DETECTION_H_
8.   #define _LCPR_PEDESTRIAN_DETECTION_H_
9.
10.  #include "basic_define.h"
11.
12.  namespace lcpr {
13.  #define MAX_PEDESTRIAN_NUM 128
14.
15.  /* Detected pedestrian information.*/
16.  typedef struct pedstrian_t {
17.      float score;
18.      LCPRBoundingBox bb;
19.  } Pedestrian;
20.
21.  /* Ouput information.*/
22.  typedef struct detection_output_t {
23.      int num;
24.      Pedestrian detected_pedestrians[MAX_PEDESTRIAN_NUM];
25.  } DetectionOutput;
26.
27.  /** @class LCPRPedestrianDetection
28.   */
29.  class PedestrianDetection {
30.
31.  public:
32.      /* Constructor and destructor.*/
33.      PedestrianDetection(void){}
34.      ~PedestrianDetection(void){}
35.
36.      /**
37.       * \brief Initialization.
38.       * \param[IN] params - the necessary parameters used for pedestrian detection.
39.       *            (It can be a pointer to a struct of paramters or a path of a file
40.       *             including all the parameters.)
41.       * \return a handle (a pointor to the inside object to achieve detection).
42.       */
43.      long initialize(void* params);
44.
45.      /**
46.       * \brief detect the pedestrian(s) in a single image.
47.       * \param[IN] handle
```

```
48.       * \param[IN] img - the rgb data of input image and its basic information.
49.       * \param[OUT] detected - detection results.
50.       * \return error code (as shown in basic_define.h).
51.       */
52.      int detect(long handle, const LCPRInputIMG& img, DetectionOutput& detected);
53.
54.      /**
55.       * \brief realse resources.
56.       * \param[IN] handle
57.       */
58.      void release(long handle);
59.  }; // PedestrianDetection
60.  }  // lcpr
61.  #endif //_LCPR_PEDESTRIAN_DETECTION_H_
```

说明：

1). 函数 initialize 用于初始化网络，指定计算资源，以及为一些必要参数赋值等（输入可以是指向“参数结构体的指针”或包含所需参数的文件路径），初始化成功返回“句柄”，即指向用于检测的实例化对象的指针，该句柄用于之后的检测和资源释放函数。

2). 函数 detect 即对输入的一帧 RGB 图像进行行人检测，这里数据格式与 OpenCV (c/c++) 相同，为 BGR,BGR,...排列方式。

3). 函数 release 用于释放资源。

(2). 属性识别

```
1.  /**
2.   * @file    attribute_recognition.h
3.   * @brief   The interface to do attribute recognition through calling the trained
4.   *          model which is provided by users.
5.   * @author  CRIPAC
6.   * @version 0.1
7.   * @date    2018/04/09
8.   */
9.  #ifndef _LCPR_ATTR_RECOGNTION_H_
10. #define _LCPR_ATTR_RECOGNTION_H_
11.
12. #include "basic_define.h"
13.
14. namespace lcpr{
15. class AttributeRecogntion {
16. public:
17.     /*Constructor and destructor.*/
18.     AttributeRecogntion(void){}
19.     ~AttributeRecogntion(void){}
20.
21.     /** \brief Initialization.
22.      *  \param[IN] params - parameters used in initialization (a pointer to a struct
```

```
23.       *             or a path about the file including all parameters).
24.       *  \return a handle (a pointor to the instantiated object to attribute recognition)
25.       */
26.      long initialize(void* params);
27.
28.      /** \brief To recognize the attributes of an image.
29.       *  \param[IN] handle
30.       *  \param[IN] img - the rgb data of input image and its basic information.
31.       *  \return recognition results.
32.       */
33.      const float* recognize(long handle, const LCPRInputIMG& img);
34.
35.      /** \brief To recognize the attributes of multiple images.
36.       *  \param[IN] handle
37.       *  \param[IN] batch_size - the number of input images.
38.       *  \param[IN] imgs - an array of images.
39.       *  \return recognition results.
40.       */
41.      const float* recognize(long handle, int batch_size, LCPRInputIMG* imgs);
42.
43.      /** \brief release the resources.
44.       *  \param[IN] handle
45.       */
46.      void release(long handle);
47.  };  // AttributeRecognition
48.  } // lcpr
49.
50.  #endif //_LCPR_ATTR_RECOGNTION_H_
```

说明：

1). 初始化函数的输入参数同“行人检测”要求。

2). 输入图像并不是完整图像，而是检测到的矩形框内对应的数据。

3). 需要显式给出输出层维数，若为一次处理多帧，请确保输出结果顺序与输入一致。

4). 需要在提供的说明文档中明确指明输出层结果与属性名称之间的对应关系。

(3). 行人再识别

```
1.   /**
2.    * @file    reid_feature_extraction.h
3.    * @brief   The interface to extract reid features through calling the trained
4.    *          model which is provided by users.
5.    * @author  CRIPAC
6.    * @version 0.1
7.    * @date    2018/04/10
8.    */
9.
10.  #ifndef _LCPR_REID_FEATURE_EXTRACTION_H_
```

```cpp
#define _LCPR_REID_FEATURE_EXTRACTION_H_

#include "basic_define.h"

namespace lcpr{
class ReIDFeatureExtraction {
public:
    /*Constructor and destructor.*/
    ReIDFeatureExtraction(void){}
    ~ReIDFeatureExtraction(void){}

    /** \brief Initialization.
     *  \param[IN] params - parameters used in ReID feature extraction (a pointer to a struct
     *             or a file path to the parameters file).
     *  \return a handle (a pointor to the instantiated object to attribute recognition)
     */
    long initialize(void* params);

    /** \brief To extract the reid features of a single image.
     *  \param[IN] handle
     *  \param[IN] img - the rgb data of input image and its basic information.
     *  \return reid features.
     */
    const float* extract(long handle, const LCPRInputIMG& img);

    /** \brief To extract reid features of multiple images.
     *  \param[IN] handle
     *  \param[IN] batch_size - the number of input images.
     *  \param[IN] imgs - an array of images.
     *  \return reid features.
     */
    const float* extract(long handle, int batch_size, LCPRInputIMG* imgs);

    /** \brief release the resources.
     *  \param[IN] handle
     */
    void release(long handle);
};  // ReIDFeatureExtraction
}   // lcpr

#endif // _LCPR_REID_FEATURE_EXTRACTION_H_
```

说明：

1). 该接口目的为提取输入图像的 ReID 特征。

2). 其他注意事项同“属性识别”：1). 2). 3). 4).

注：

1). 请依据上述接口给出可正常运行的单机版测试代码。

2). 目前系统平台支持 **Caffe(2)，Pytorch，Tensorflow(keras)**以及 **mxnet** 框架训练的模型，网络结构中不能包含自定义的 **layer**，否则请提供源代码，或封装好的动态链接库。

3). 若为 **python** 代码，请形式上与 **C++**接口保持一致，即包含前述的三个函数即可，无需再另裹一层 **C++**，此外应给出详细说明文档，包括依赖环境和版本等。

4). 若提供源代码请确保代码内部无“内存泄露”情况，建议使用 **valgrind** 自行进行检测。

5). 目前只能调用在 **linux** 环境下封装的代码。

6). 集群环境中 **cuda** 版本为 **8.0**，**cudnn** 为 **6.0.**