**基于Lire的分析与改进**

**改进与展示实现方案**

Version 1.0

小组成员：

刘少凡

宋昱材

吴沂楠

黄飞

**版本变更记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 变更时间 | 修改人 | 审核人 | 备注 |
| 1.0 | 20170426 | 宋昱材 | 刘少凡 吴沂楠 黄飞 | 初稿 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1改进目标 4](#_Toc480999820)

[2工作内容 4](#_Toc480999821)

[2.1 Lire源码修改 4](#_Toc480999822)

[2.2 Caffe环境配置和模型调用 4](#_Toc480999823)

[2.4 JNI实现 5](#_Toc480999824)

[2.3界面实现 5](#_Toc480999825)

[3实现方案 6](#_Toc480999826)

[3.1 Lire源码修改 6](#_Toc480999827)

[3.2 Caffe环境配置和模型调用 6](#_Toc480999828)

[3.3 JNI实现 7](#_Toc480999829)

[3.2.1 Java端 7](#_Toc480999830)

[3.2.2 C++端 7](#_Toc480999831)

[3.4 界面实现 7](#_Toc480999832)

# 1改进目标

作为一种开源框架，Lire的代码对开发者完全透明，程序代码具备简明、方便和清晰的构架设计与函数接口来方便用户的使用。而且，Lire的实现框架使得开发者可以方便地引入新的技术、算法或模块，以满足开发人员的不同需求。因此Lire具备良好的可修改性或可扩展性。

项目的改进目标基于上述的可修改性。

项目计划在Lire中增加一种新的特征提取方法——CNN特征。CNN（Convolutional Neural Network），即卷积神经网络。项目计划利用一个已训练好的面向图像分类任务的CNN模型，将其作为特征提取工具，从模型中提取某一层输出作为图像特征。

这种尝试的出发点基于实际开发中时常会出现的场景，即CBIR系统开发者计划使用Lire工具包进行系统开发，但Lire工具包中并未实现开发者所希望使用的图像特征，因此需要向工具包中扩展该图像特征。项目站在CBIR系统开发者角度，对Lire针对特定开发目标进行扩展。

# 2工作内容

## 2.1 Lire源码修改

Lire的特征提取方法的具体实现在imageanalysis包中，通过LireFeature接口定义了特征提取类需要实现的方法，具体的特征提取类如CEDD等通过继承该接口进行具体的实现。因此需要通过继承LireFeature实现新的CNN类。

另外，要在图像入库和图像检索模块增加CNN特征类的接口。

## 2.2 Caffe环境配置和模型调用

深度学习框架Caffe的运行环境需要依赖多种工具和库的支持，因此需要对操作系统的环境进行配置。

编写C++代码，实现对模型的调用。C++代码主要实现两个功能，一是读取模型配置文件和参数文件将模型加载到内存中；二是将需要提取特征的图片输入模型，获得模型输出结果。

## 2.4 JNI实现

由于Lire基于Java实现，Caffe模型调用基于C++实现，因此需要利用java的JNI机制实现Lire对Caffe的调用。

## 2.3界面实现

前端界面主要包括图像主界面、图像入库界面、图像检索界面和检索结果界面。

# 3实现方案

## 3.1 Lire源码修改

imageamalysis包中的LireFeature接口定义了图像特征类需要实现的方法，具体内容如下所示：

|  |
| --- |
| **package** net.semanticmetadata.lire.imageanalysis;  **import** java.awt.image.BufferedImage;  **public** **interface** LireFeature **extends** Histogram {  //获取特征名称  **public** String getFeatureName();  //获取字段名  **public** String getFieldName();  //提取特征  **public** **void** extract(BufferedImage image);  //获取图像特征的Byte表示  **public** **byte**[] getByteArrayRepresentation();  //重置图像特征  **public** **void** setByteArrayRepresentation(**byte**[] featureData);  //有位移和长度的重置图像特征  **public** **void** setByteArrayRepresentation(**byte**[] featureData, **int** offset, **int** length);  //获取图像特征  **public** **double**[] getDoubleHistogram();  //获取该特征与输入特征的距离  **float** getDistance(LireFeature feature);  //获取特征的字符串表示  java.lang.String getStringRepresentation();  //以字符串重置图像特征  **void** setStringRepresentation(java.lang.String featureVector);} |

通过继承LireFeature接口，实现CNN类。

另外，在Lire工具包的DocumentBuilder.java、DocumentBuilderFactory.java、ImageSearcherFactory.java和ImageSearcher.java中增加相应的构造方法实现和字段添加，实现对索引构造、特征存储和特征搜索代码的调用。

## 3.2 Caffe环境配置和模型调用

项目计划在Ubuntu 14.04操作系统上配置Caffe，依据Caffe文档依次安装依赖库，实现Caffe的安装。

通过C++代码调用Caffe的相关接口实现加载模型和调用模型的相关功能。

## 3.3 JNI实现

### 3.2.1 Java端

Java端实现GetCNN类，通过调用native的C++方法实现对Caffe模型的调用。GetCNN类应当包含以下成员变量和方法：

**表 3.1 GetCNN类成员变量与方法作用**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 变量/方法 | 类型 | 作用 |
| NetTxt | 成员变量 | String | 记录CNNs模型定义文件路径 |
| NetPara | 成员变量 | String | 记录CNNs模型参数文件路径 |
| LayerName | 成员变量 | String | 记录CNNs模型层名 |
| loadCNN | 私有方法 | void | JNI调用本地Caffe代码实现CNNs模型加载 |
| getCNN | 私有方法 | float[] | JNI调用本地Caffe代码实现特征提取 |
| getFeature | 公共方法 | flaot[] | 共外部调用的图像特征提取接口 |

其中loadCNN和getCNN为native方法，通过调用本地C++代码实现。通过javah命令生成C++所需的.h头文，供C++代码实现时包含。

### 3.2.2 C++端

C++代码包含java生成的头文件，实现头文件中定义的具体方法，将代码生成动态链接库，供java代码调用。

## 3.4 界面实现

前端界面主要包括程序主界面、入库界面、图像检索界面和检索结果界面。

用户操作程序界面的流程如图3.1所示。



图 3.1 用户操作界面流程图

界面相关类信息如表3.2所示。

**表 5.3 界面相关类信息**

|  |  |  |
| --- | --- | --- |
| 类 | 界面 | 功能描述 |
| StartFrame | 主界面 | 提供入库与检索不同功能的选择 |
| StorageFrame | 入库界面 | 选择入库图片路径，进行图片入库 |
| MainFeame | 检索界面 | 选择检索图片，进行图像检索 |
| ImageResultFrame | 检索结果界面 | 检索结果显示 |