# 上海大学 2019 ～ 2020 学年

# 课程报告成绩评价表

课程名称：《模式识别》课程编号：08306089

报告名称：基于 SVM 处理喉癌 CT 图像分类问题

姓　　名：荀嘉皓　　学　　号：17121466

报告评语：

| |
|---|
| |

报告成绩：

| 方案设计（20 分） | | 实验验收（20 分） | | 书面报告（60 分） | | | 总分 |
|---|---|---|---|---|---|---|---|
| 可行性（10 分） | 创新性（10 分） | 规范性（10 分） | 演示效果（10 分） | 规范性（20 分） | 完整性（20 分） | 科学性（20 分） | |
| | | | | | | | |

任课教师:

评阅日期:　　年　　月　　日

附件 1:书面报告

# 基于 SVM 处理喉癌 CT 图像分类问题

荀嘉皓(17121466)

**摘要：**在医学诊断领域,CT 图像的自动分类能有效地提高医学诊断的准确率和速率。为了探究喉癌 CT 图像以及其对应病人治疗后情况的联系，本文采取模式识别的方法尝试对 CT 图像分类。考虑到 CT 图片样本太少不太合适 CNN 等神经网络算法，故本文尝试通过 HOG、LBP、SIFT 图片特征提取方法以及传统 SVM 相结合方法来对 CT 图像进行分类。实验中发现 HOG 方法结合线性 SVM 的方法正确率较为可观，平均正确率为 78%

## 1 引言

### 1.1 提出问题

#### 1.1.1 课题研究背景和意义

随着大量新型的医疗设备应用于临床，医学图像的种类越来越多，仅靠人为诊断无疑会给医生带来繁重的工作，因此，研究医学图像的识别技术势在必行。

医学 CT 图像分类技术的研究有以下三方面的意义：（1）在保证精度的前提下自动处理大量图片数据，能有效地提高医学诊断的准确率和速率且不受人为因素影响；（2）数字图像处理和模式识别技术在不断发展，使实现快速准确的医学图像处理的成本不断下降；（3）利用智能信息系统可以实现不同地区临床信息的沟通。模式识别技术在提高诊断准确率的同时也降低了医疗成本[1]。

#### 1.1.2 课题技术难点

由于医学 CT 图像具有不同于其他一般图像的特征，如细节纹理较多，分辨率较低，样本量较小，并且要严格保证识别分类的准确性。因此，需要寻找一些适用于医学图像的特征提取算法以及模式识别方法，来对其进行识别分类。

### 1.2 求解方案分析

对于图像预处理，考虑到医学图像的特殊性，采用图片锐化可能会产生冗余干扰信息，如图 1 所示，出现了不必要的边缘信息。因此数据预处理阶段仅考虑 ROI 区域提取[2]和 gamma 变换。其中，ROI 区域的提取方法采用的是掩膜法，即只保留标注位置的原始像素值，其余位置的像素值置为 0，这样可以免去非病变区域带来的影响。而通过 gamma 变换可以提升 CT 图像暗部的细节，方便特征提取

此外，由于样本数量比较稀少，预处理后合计有效图片 207 张，故考虑采取传统的特征提取方法和 SVM 分类器进行实验。图片特征主要考虑两个方面，一个是图片的面特征信息，另一个是图片的点特征信息。针对面特征信息，所采用的方法是 HOG 方法以及 LBP 方法。对于点特征信息，所采用的是 SIFT 方法。由于这是一个二分类问题，且无法事先预测 CT 图像特

征的线性可分性，故 SVM 核函数考虑三种常用的核函数（线性核、高斯核以及多项式核）对样本进行训练和测试。

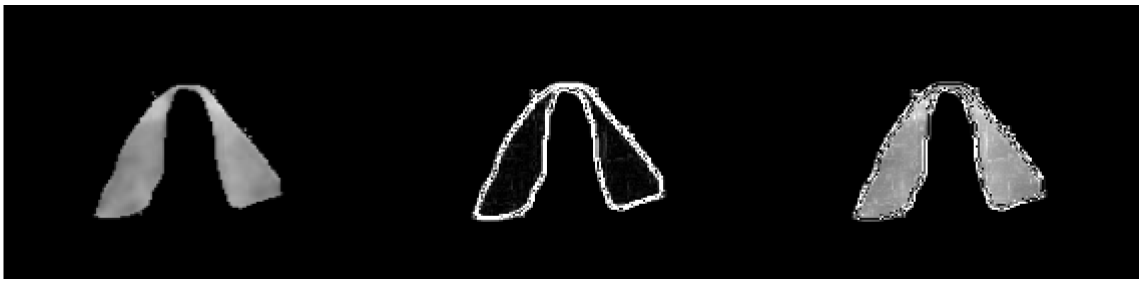


图 1　图像锐化冗余信息示例

### 1.3　论文概述

本文首先介绍了三种特征提取算法（HOG、LBP、SIFT）的相关原理和实现步骤，并结合样本进行了可视化。其次，描述了 CT 图像分类算法的实现，包含了算法整体框架和相关改进。算法整体框架包含了上述三种特征提取算法，以及从图片读入到样本训练的整个流程。相关改进中包含了对于原算法的部分优化。然后是实验描述部分，包含了实验数据、实验方案设计和实验结果分析三个部分。实验数据中包含了数据来源、存储格式和数量。实验设计中包含了对于特征提取和 SVM 核函数选择的相关实验。实验分析包含了对实验结果的具体评价。最后是实验结论和个人体会部分。

## 2　相关算法概述

### 2.1　HOG 特征提取算法

#### 2.1.1　梯度计算[3]

由于在目标边缘处灰度变化较大，因此，在边缘处灰度的梯度就较为明显，所以，梯度能够更好的表征目标的特征。此外，由于图像数值是离散的，故可以利用一阶差分代替微分求离散图像的梯度大小和梯度方向，计算得到水平方向和垂直方向的梯度分别如下

$$G_h(x,y) = f(x+1,y) - f(x-1,y), \forall x,y \tag{1}$$

$$G_v(x,y) = f(x,y+1) - f(x,y-1), \forall x,y \tag{2}$$

可以得到梯度值(梯度强度)和梯度方向分别为

$$M(x,y) = \sqrt{G_h(x,y)^2 + G_v(x,y)^2} \tag{3}$$

$$\theta(x,y) = \arctan\left(G_h(x,y)/G_v(x,y)\right) \tag{4}$$

#### 2.1.2　单元划分

计算得到梯度的幅值和梯度方向之后，紧接着就是要建立分块直方图，得到图像的梯度大小和梯度方向后根据梯度方向对图像进行投影统计，首先将图像划分成若干个块(Block)，每个块又由若干个细胞单元(cell)组成，细胞单元由更小的单位像素(Pixel)组成，然后在每个细胞单元中对内部的所有像素的梯度方向进行统计。

#### 2.1.3　区块选择及其归一化

为了应对光照和形变，梯度需要在局部进行归一化。常用的区块有两种，分别是矩形区块和圆形区块。每个方格内对像素梯度方向进行统计可以得出一个特征向量，一个区块内有多个方格，也就有多个特征向量。可以用 L2-norm 归一化，归一后的特征向量为，

$$v = \frac{v}{\sqrt{||v||_2^2 + \varepsilon^2}} \tag{5}$$

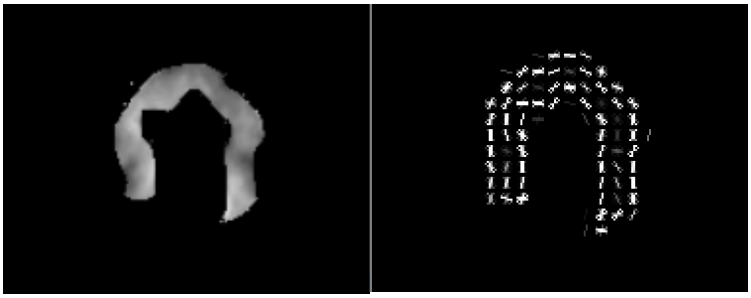


图 2　HOG 特征图示例

## 2.2　LBP 特征提取算法

### 2.2.1　LBP 算法描述[4]

原始的 LBP 算子定义为在 3x3 的窗口内，以窗口中心像素为阈值，将相邻的 8 个像素的灰度值与其进行比较，若周围像素值大于中心像素值，则该像素点的位置被标记为 1，否则为 0。这样，3x3 邻域内的 8 个点经比较可产生 8 位二进制数（通常转换为十进制数即 LBP 码，共 256 种），即得到该窗口中心像素点的 LBP 值，并用这个值来反映该区域的纹理信息。

### 2.2.2　LBP 算法步骤

1） 首先将检测窗口划分为 16×16 的小区域（cell）。

2） 对于每个 cell 中的一个像素，将相邻的 8 个像素的灰度值与其进行比较，若周围像素值大于中心像素值，则该像素点的位置被标记为 1，否则为 0。这样，3x3 邻域内的 8 个点经比较可产生 8 位二进制数，即得到该窗口中心像素点的 LBP 值。

3） 然后计算每个 cell 的直方图，即每个数字（假定是十进制数 LBP 值）出现的频率；然后对该直方图进行归一化处理。

4） 最后将得到的每个 cell 的统计直方图进行连接成为一个特征向量，也就是整幅图的 LBP 纹理特征向量。

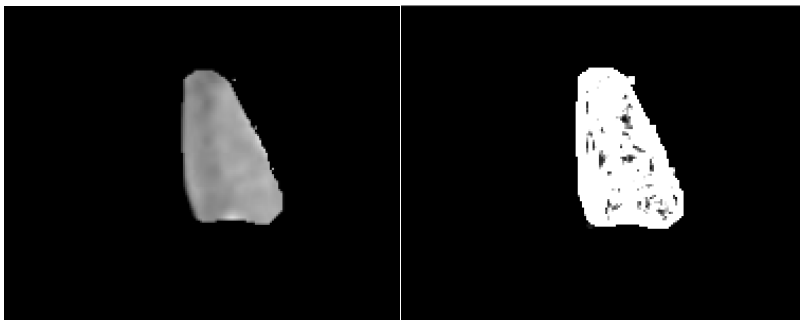


图 3　LBP 特征图示例

## 2.3　SIFT 算法特征提取

### 2.3.1　尺度空间极值检测[5]

通过使用高斯差分函数来计算并搜索所有尺度上的图像位置，用于识别对尺度和方向不变的潜在兴趣点。在实际计算时，使用高斯金字塔每组中相邻上下两层图像相减，得到高斯差分图像（DoG）。

### 2.3.2　关键点定位

通过一个拟合精细的模型在每个候选位置上确定位置和尺度，关键点的选择依赖于它们的稳定程度。我们可以通过拟合三维二次函数来精确确定关键点的位置和尺度，同时去除低对比度的关键点和不稳定的边缘响应点(因为 DoG 算子会产生较强的边缘响应)，以增强匹配稳

定性、提高抗噪声能力。

2.3.3　方向匹配

基于局部图像的梯度方向，为每个关键点位置分配一个或多个方向，后续所有对图像数据的操作都是相对于关键点的方向、尺度和位置进行变换，从而这些变换提供了不变形。

2.3.4　关键点描述

在每个关键点周围的区域内以选定的比例计算局部图像梯度，这些梯度被变换成一种表示，这种表示允许比较大的局部形状的变形和光照变化。

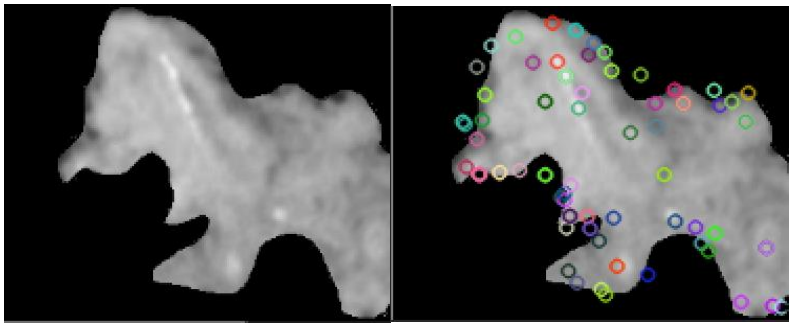


图 4　SIFT 特征点图示例

# 3　算法实现描述

算法主要采用了三种特征提取方法并结合 SVM 分类器来对预处理过的 CT 图像进行分类。考虑到医学图像的特殊性，图片预处理部分仅采用了掩膜和 gamma 变换，而未使用锐化技术。特征提取方法分别为 HOG、LBP 和 SIFT，并分别测试了 SVM 线性核函数和非线性核函数的准确率。

## 3.1　算法总体框架

在图片预处理过程中，我们首先对所有图像通过插值的方法，缩放到相同的大小。然后通过掩膜的方法提取 ROI 区域，即只保留标注位置的原始像素值，其余位置的像素值置为 0，这样可以去除非病变区域带来的影响。

对于 HOG 方法而言，我们首先读入预处理过的图片，然后通过 HOG 方法（先计算图像梯度，然后将图片划分成若干个区域并对区域内的梯度值进行统计，最后将这些梯度向量归一化形成特征向量）提取特征向量。之后，我们可以对这些特征向量通过 PCA 方法进行降维，以提高训练和识别速度。最后，将降维后的特征向量放入 SVM 中训练即可。


图 5　HOG+SVM 算法框架图

对于 LBP 方法而言，我们首先读入预处理过的图片，然后通过 LBP 方法（先将图片划分成若干个小区域，然后在每个小区域内将每个像素值与其相邻的 8 个像素值作比较，产生的二进制数作为该点的 LBP 值，最后统计每个区域 LBP 值出现频率的直方图并归一化得到特征向量）。最后将特征向量放入 SVM 中训练即可。

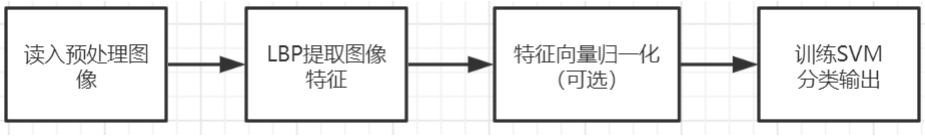



图 6　LBP+SVM 算法框架图

对于 SIFT 方法而言，我们首先读入预处理过的图片，然后利用 SIFT 算法（先对图片进行尺度极值检测，然后进行关键点定位和方向匹配，最后进行关键点描述）从预处理过的图像中提取视觉词汇向量，这些向量代表的是图像中局部不变的特征点。其次，将所有特征点向量集合到一块，利用 K-Means[6]算法合并词义相近的视觉词汇，构造一个包含 K 个词汇的单词表。之后，通过 BOW 方法统计单词表中每个单词在图像中出现的次数，从而将图像表示成为一个 K 维数值向量并归一化。最后将特征向量放入 SVM 中训练即可。

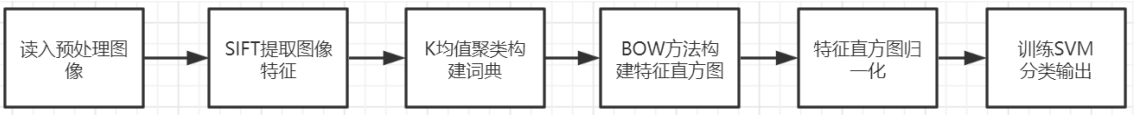



图 7　SIFT+SVM 算法框架图

### 3.2　PCA 降维及其改进分析

将总数据集（合计 207 张图片）按 70%（144 张）和 30%（63 张）的比例随机划分训练集和样本集，对 HOG、LBP、SIFT 三种方法分别进行单组测试，结果如下表所示。

表 1　三种特征提取方法在测试集单次测试的特征维数、正确率及耗时对比图

| 方法名称 | 特征向量维数 | 正确率 | 耗时/s |
|---|---|---|---|
| HOG+SVM | 57600 | 88.9% | 2.0975 |
| LBP+SVM | 256 | 68.3% | 1.2016 |
| SIFT+SVM | 100 | 71.4% | 0.0538 |

我们可以从上表中发现，HOG 方法所提取的特征向量维数是最高的，且耗时也是最长的，相对的正确率也是最高的。LBP 方法所提取的特征向量维数相对较少，但耗时也较长，但正确率是最低的。SIFT 方法的特征向量维数取决于 k-means 方法的 k 值的选择，这边选择的是 100 维，可发现该方法维数较低，耗时极少，且准确率较为良好。

考虑到 HOG 方法准确率较高，但提取的特征维数也很高，会导致之后的训练和识别分类过程变慢，故尝试采用 PCA 主成分分析方法[7]对该高维特征进行降维处理。

表 2　HOG 方法在测试集单次测试的特征维数、正确率及耗时对比图

| 特征向量维数 | 正确率 | 耗时/s |
|---|---|---|
| 57600 | 88.9% | 2.0975 |
| 207(样本个数) | 88.9% | 0.0678 |
| 100 | 87.3% | 0.0538 |
| 50 | 82.5% | 0.0548 |
| 10 | 74.6% | 0.0558 |

从上表中我们可以发现，当特征向量维数远大于样本个数的时候，我们可以借助 PCA 将维数将至样本个数以内。此外，当特征向量维数等于样本个数时正确率是最高的，且耗时也较为可观，但随着维数的进一步减少，正确率也会随之减少，而耗时减小幅度不大。考虑到分类正确性的重要性，取样本个数作为维数较为合理。

综上而言，利用 PCA 方法对特征向量的降维是一种有效的减少计算机开销的一种方法。但其缺点在于，需要读取整个训练集和测试集一起降维，实际应用中难以实现。

### 3.3 BOW 方法改进 SIFT 特征提取及其分析

对于传统的 SIFT 而言，我们所得到的是所有图像的特征点以及 n x 128 维特征(n 为特征点个数)，而每个图像的特征点数目不一定相同。由于数目不相同，我们就不能直接将其作为图片特征进行训练。因此我们需要借助 K-Means 和 BOW（Bag-of-words）模型，将每一张图像的特征点转换为维数相同的特征向量。

K-Means 算法是一种无监督的聚类算法。其算法思想大致为：先从样本集中随机选取 k 个样本作为簇中心，并计算所有样本与这 k 个“簇中心”的距离，对于每一个样本，将其划分到与其距离最近的“簇中心”所在的簇中，对于新的簇计算各个簇的新的“簇中心”。

Bag-of-words 模型是信息检索领域常用的文档表示方法。在信息检索中，BOW 模型假定对于一个文档，忽略它的单词顺序和语法、句法等要素，将其仅仅看作是若干个词汇的集合，文档中每个单词的出现都是独立的，不依赖于其它单词是否出现。也就是说，文档中任意一个位置出现的任何单词，都不受该文档语意影响而独立选择的[8]。

为了表示一幅图像，我们可以将图像看作文档，即若干个“视觉词汇”的集合，同样的，视觉词汇相互之间没有顺序。因此我们可以用 SIFT 算法从图像中提取不变特征点，作为视觉词汇，并利用 K-Means 构造单词表，用单词表中的单词表示一幅图像。最后，统计单词表中每个单词在图像中出现的次数，从而将图像表示成为一个 K 维数值向量。归一化后，可作为图片的特征向量。

## 4 实验描述

### 4.1 实验数据和实验方案

#### 4.1.1 实验数据介绍

实验数据文件夹有预后好和预后差两个文件夹，每个文件内又有以 “病人编号,肿瘤分期数,癌症位置”此格式命名的若干子文件夹。子文件夹中包含病人的 CT 图像以及对应的喉癌位置标注图片。合计图片 445 张，有效图片（去除 MRI 图像和无位置标注图像）为 414 张。经过掩膜预处理后，总样本集为 207 张。


图 8 文件夹格式示例

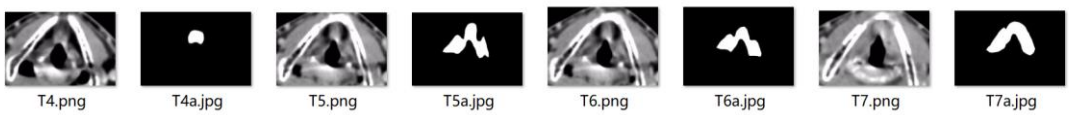



图 9 图片样本格式示例

#### 4.1.1 实验方案设计

本文的算法框架主要包含两大部分，其一是图片特征向量提取，其二是 SVM 分类器训练。为了探究不同特征提取方法，以及不同 SVM 核函数(线性和非线性)选择下的分类效果，设计

了一下两种实验。

1）SVM 核函数选择实验：

在相同特征提取算法下，比较不同核函数所造成的正确率和分类结果差异。

2）特征提取算法正确率差异实验：

在相同核函数下，比较不同特征提取算法选择所造成的正确率和分类结果差异。

### 4.2 SVM 核函数选择实验及结果分析

在 SVM 中核函数的作用是解决了其无法处理非线性可分的特征向量的问题。而 CT 图像的形态特征是通过图像特征提取算法获得的。因此我们无法事先得知 CT 图像的特征向量的分布状态，故我们需要不断调整核函数类型，以使得训练出的模型鲁棒性和准确性较高。本实验总共测试了三个比较常见的核函数，分别为线性核函数、高斯核函数以及多项式核函数。

测试形式为，将总数据集（合计 207 张图片）按 70%（144 张）和 30%（63 张）的比例随机划分训练集和样本集，对 HOG、LBP 和 SIFT 三种特征提取方法测试 10 轮后计算正确率的均值。

表 3 不同核函数下准确率比较

| 方法名称 | 平均正确率（线性） | 平均正确率（高斯） | 平均正确率（多项式） |
|---|---|---|---|
| HOG+SVM | 77. 6% | 65. 1% | 60. 6% |
| LBP+SVM | 68. 4% | 63. 9% | 67. 4% |
| SIFT+SVM | 69. 8% | 63. 4% | 59. 4% |

根据上表我们可以发现，当算法一定时，线性核函数所训练出的模型的准确性高于高斯核函数所训练出的模型；高斯核函数所训练出的模型的准确性高于多项式核函数所训练出的模型。而当核函数一定时，HOG 方法的正确率高于 LBP 和 SIFT 方法，而 LBP 和 SIFT 方法正确率相似。

由此我们可以发现，在大多数情况下，线性核函数所训练出的模型准确率要高于非线性核函数。由此我们可以推测，CT 图像的特征向量可能是具有一定线性可分性的。后续的一些实验也是基于线性 SVM 基础上设计的，

### 4.3 特征提取算法正确率差异实验及结果分析

在 4.2 中我们已经发现线性核 SVM 的分类效果要好于非线性 SVM。于是，我们需要在选用线性核 SVM 的基础上，挑选一个比较合适的特征提取方法，最好能达到特征向量线性可分的效果，以提高分类正确率。

将总数据集（合计 207 张图片）按 70%（144 张）和 30%（63 张）的比例随机划分训练集和样本集，对 HOG、LBP 和 SIFT 三种特征提取方法测试 100 轮后并将其画出正确率变化并计算正确率的均值，SVM 核函数采用线性核函数。

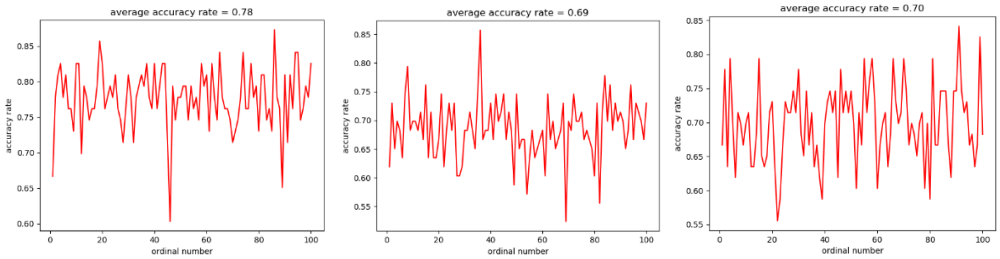



图 10 平均正确率对比图（从左至右分别为 HOG、LBP、SIFT）

从上图中我们可以发现三种方法准确率波动都较大，这就说明 SVM 分类器对样本集的依

赖比较强。若训练的样本种类不充足的话，会导致 SVM 训练出的模型泛化能力较差。此外，当 SVM 核函数一定的时候，HOG 方法的准确率是最高的，波动也更为稳定。

**表 4　测试集单次测试结果对比表**

| 方法名称 | accuracy | precision | recall | f1 |
|---|---|---|---|---|
| HOG+SVM | 76.19% | 85.71% | 60.0% | 70.59% |
| LBP+SVM | 73.01% | 59.09% | 61.90% | 60.46% |
| SIFT+SVM | 77.78% | 62.06% | 85.71% | 72.00% |

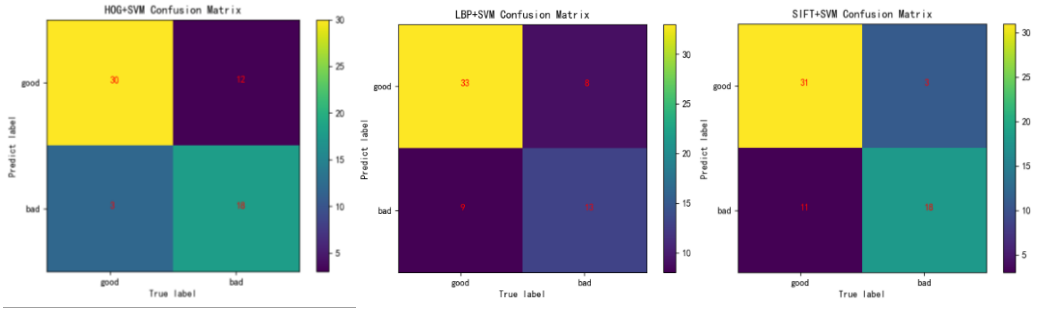



图 11　混淆矩阵对比图（从左至右分别为 HOG、LBP、SIFT）

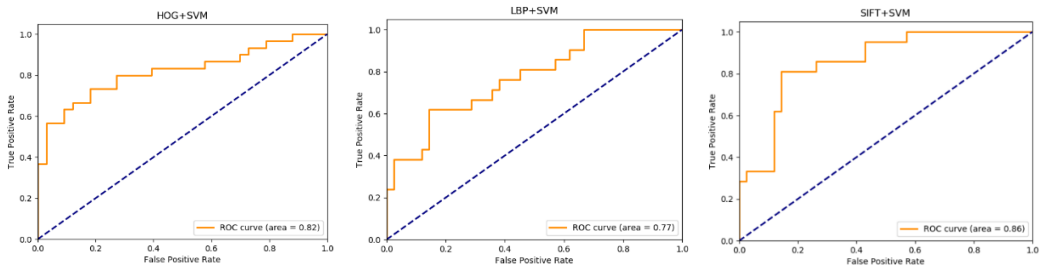



图 12 ROC 对比图（从左至右分别为 HOG、LBP、SIFT）

在表 4、图 11 和图 12 我们可以发现，三种基于 SVM 方法的正确率大多介于 69%-80%之间。HOG 方法的优势在于总体正确率和准确率都比较高, ROC 曲线面积为 0.82 较为可观; SIFT 方法的优势在于 recall 值和 f1-score 值较高，且将预后 bad 标签判断为预后 good 标签的情况比较少，比较适用于医疗环境, ROC 面积为 0.86，较为可观; 而 LBP 方法相比较而言，虽然它将预后 bad 标签判断为预后 good 标签的情况比较少，但其正确率和准确率比较低，除此以外就没有太出彩地方，ROC 面积也仅有 0.77，不是很理想。

## 5　结论

由于实验样本比较稀少，本文考虑通过 SVM 进行小样本的分类。在预处理阶段，采用的是掩膜的方法，只保留标注位置的原始像素值，其余位置的像素值置为 0，从而达到 ROI 提取的效果。此外，我本文选择了三种图片特征提取的方法，分别为 HOG、LBP 和 SIFT，其中 HOG 和 LBP 方法表达的是图片面特征，而 SIFT 表达的是图片的点特征。

从分类正确性和稳定性来看，我们可以从图 10 发现三种方法准确率波动都较大，这就说明 SVM 分类器对样本集的依赖比较强。若训练的样本种类不充足的话，会导致 SVM 训练出

的模型泛化能力较差。此外，当 SVM 核函数一定时，HOG 方法的准确率是最高的，波动也更为稳定。

再从 3.2 以及 4.2 和 4.3 两个实验中综合来看，我们可以推测图像特征是具有一定线性可分性的（三种特征提取方法在线性 SVM 上的表现优于非线性 SVM）。其次，从综合性能上来看，HOG 特征提取结合线性 SVM 有着较为可观的分类效果，且在经过 PCA 主成分分析法降维后有着较为不错的分类速度。对于 SIFT 结合线性 SVM 方法而言，虽然它的准确率不及 HOG 方法，但其 ROC 曲线面积相对较大，且将预后 bad 标签判断为预后 good 标签的情况比较少，比较适用于医疗环境。相比之下 LBP 方法就逊色不少，虽然它的平均正确率与 SIFT 相似，但是它分类图片所用时间比 SIFT 方法要久很多。

# 6 学习体会和建议

学习模式识别已经快一个学期了，可以说模式识别这门课程，是我大学三年以来感觉最为晦涩难懂的一门课。它的难点不仅在于实践编程上面，更在于理解算法背后的数学原理。可以说模式识别是建立在数学之上的，是数学的一种实际应用。此外，学习内容的跨度也是比较大的，从最初比较传统的分类方法，如贝叶斯决策、SVM 等，到比较现代的深度学习方法等。

虽然由于数学基础的原因，课堂上讲述的内容确实很难一下就理解。但它更多的是告诉你模式识别中究竟有哪些内容，在之后的学习中就不用盲目地寻找接下来该学习哪些东西。此外，课堂研讨和实验上机的结合，不仅能够锻炼队内合作协调意识，也确实提高了自己的动手能力和知识储备。

最后，模式识别这门课程还能与其他课程相互结合学习，达到共同进步的效果，如本文中采用的部分特征提取方法就是从数字图像处理课堂中学到的。由此，就更能够提升我对模式识别和数字图像处理课程的学习兴趣了。

**参考文献：**

[1]刘雪鸥. 医学图像模式识别技术的研究及应用[D]. 太原理工大学, 2016.

[2]孙伟, 王小伟, 游世军, 石昊坤, 胡艳辉. 模式识别在医用超声数字图像特征提取中的应用研究[J]. 中国医学装备, 2020, 17(02): 1-5.

[3]Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]//2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). IEEE, 2005, 1: 886-893.

[4] Liao S, Zhu X, Lei Z, et al. Learning multi-scale block local binary patterns for face recognition[C]//International Conference on Biometrics. Springer, Berlin, Heidelberg, 2007: 828-837.

[5] Lowe D G. Distinctive image features from scale-invariant keypoints[J]. International journal of computer vision, 2004, 60(2): 91-110.

[6]Jain A K. Data clustering: 50 years beyond K-means[J]. Pattern recognition letters, 2010, 31(8): 651-666.

[7] Price A L, Patterson N J, Plenge R M, et al. Principal components analysis corrects for stratification in genome-wide association studies[J]. Nature genetics, 2006, 38(8): 904-909.

[8]Issolah M, Lingrand D, Precioso F. SIFT, BoW Architecture and one-against-all Support Vector Machine[C]//CLEF (Working Notes). 2013.

附件 2：自学报告讲义

2020/3/27

## 什么是线性回归

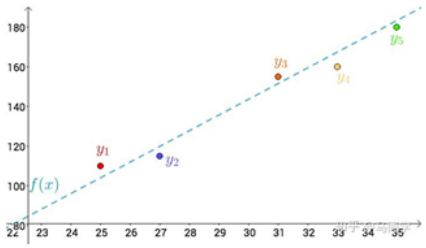
比如已知有关温度与冰淇淋的销量的一些数据：

| | 销量 |
|---|---|
| 25° | 110 |
| 27° | 115 |
| 31° | 155 |
| 33° | 160 |
| 35° | 180 |

我们该如何求解它们之间的对应关系？

19

将已有数据在笛卡尔系坐标中画出：




我们可以猜测它们之间存在某种线性关系

即 f(x) = ax + b

20

## 最小二乘法的目的：

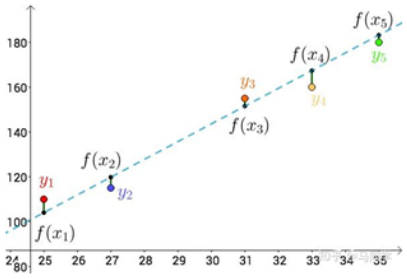
寻找到一个形如 $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_n x_n$，$\theta_i$为参数的最优预测函数，使得其对于已有的数据**误差**最小。

$$h_\theta(\mathbf{x}) = \mathbf{X}\theta$$

注：为了便于理解，可以先考虑只有一个特征的情况 $h_\theta(x) = \theta_0 + \theta_1 x$

21

## 如何定义误差




设直线方程为：$h_\theta(x) = \theta_0 + \theta_1 x$

损失函数 = $\sum$ (理论值 − 观测值)$^2$

即，$J_\theta(x) = \frac{1}{2}\sum_{i=1}^{n}(h_\theta(x_i) - y_i)^2$

22

## 最小二乘的求解过程

设 $h_\theta(x) = \theta_0 + \theta_1 x$ 的矩阵表达式为 $h_\theta(\mathbf{x}) = \mathbf{X}\theta$ ，其中X为样本输入向量

则损失函数 $J_\theta(x) = \sum_{i=1}^{n}(h_\theta(x_i) - y_i)^2$

可变形为：$J(\theta) = \frac{1}{2}(\mathbf{X}\theta - \mathbf{Y})^T(\mathbf{X}\theta - \mathbf{Y})$ ，其中Y为样本输出向量，θ为参数向量

将损失函数对θ向量求偏导后，结果取0：$\frac{\partial}{\partial\theta}J(\theta) = \mathbf{X}^T(\mathbf{X}\theta - \mathbf{Y}) = 0$

解得最终结果为：$\theta = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$

注：X 为 m x n 维的矩阵 (m代表样本数)
θ 为 2 x 1 维的列向量
Y 为 m x 1 维的列向量
Xθ-Y 为 m x 1 维的列向量

25

## 误差函数的正确性证明（略）

根据中心极限定理，误差应满足正态分布

1) $y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)}$

$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(\epsilon^{(i)})^2}{2\sigma^2}}$

$p(y^{(i)}|x^{(i)};\theta) = \frac{1}{\sqrt{2\pi}\sigma}e^{(-\frac{(y^{(i)}-\theta^T x^{(i)})^2}{2\sigma^2})}$

⇒

2) $L(\theta) = \prod_{i=1}^{m} p(y^{(i)}|x^{(i)};\theta)$

$= \prod_{i=1}^{m}\frac{1}{\sqrt{2\pi}\sigma}e^{(-\frac{(y^{(i)}-\theta^T x^{(i)})^2}{2\sigma^2})}$

似然函数

3) $\ell(\theta) = logL(\theta)$

$= log\prod_{i=1}^{m}\frac{1}{\sqrt{2\pi}\sigma}e^{(-\frac{(y^{(i)}-\theta^T x^{(i)})^2}{2\sigma^2})}$

⇒ $= \sum_{i=1}^{m}\frac{1}{\sqrt{2\pi}\sigma}e^{(-\frac{(y^{(i)}-\theta^T x^{(i)})^2}{2\sigma^2})}$

$= mlog\frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2}\cdot\frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$

对数似然函数

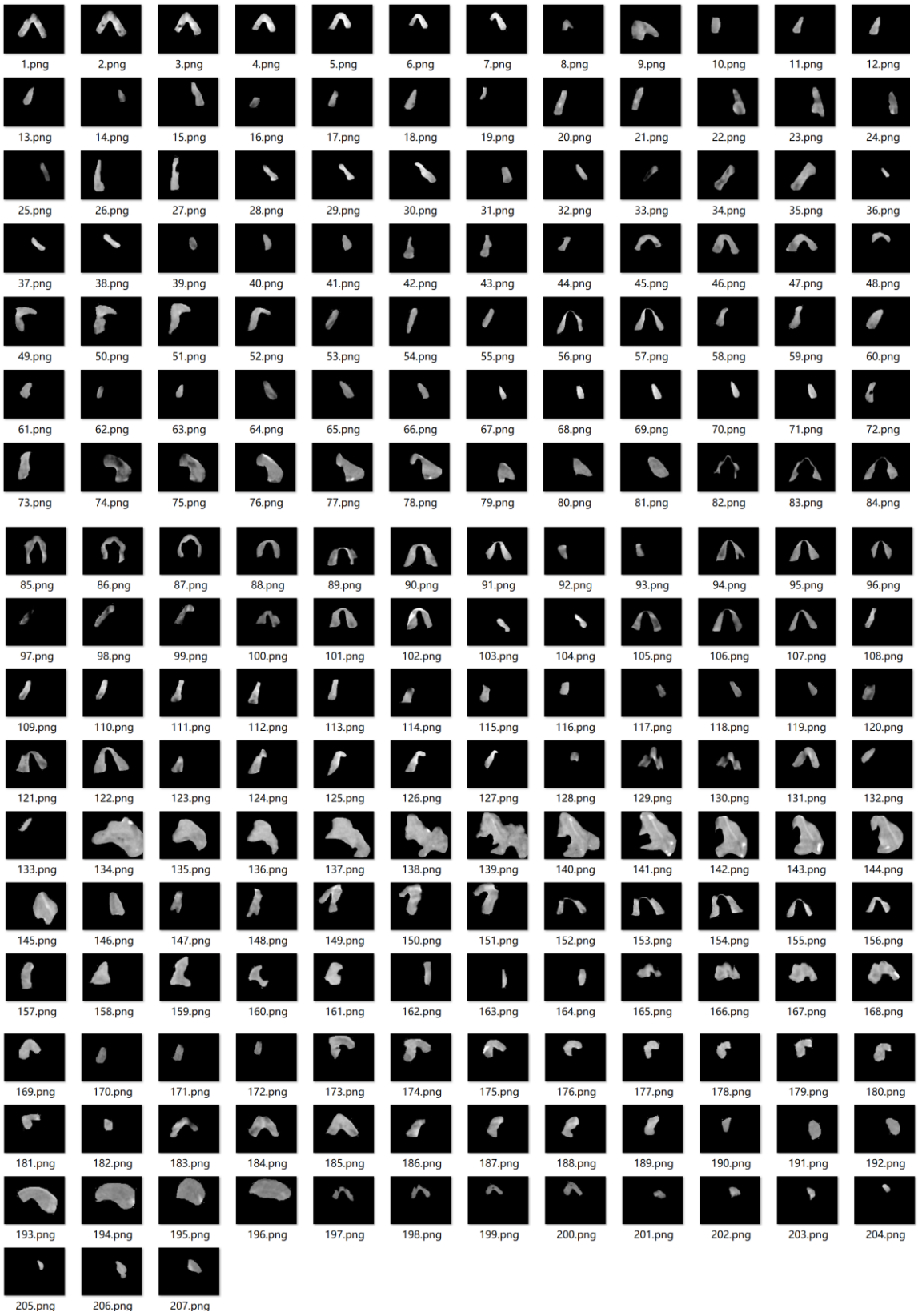
⇒

4) $J_\theta(x) = \sum_{i=1}^{n}(h_\theta(x_i) - y_i)^2$ 必须最小

26

附件 3：核心算法的主要源码和数据样例

数据样例（预处理过后）：



核心代码：

```
#HOG 特征提取：
fd = hog(image, orientations=12, block_norm='L1', pixels_per_cell=[8, 8],
cells_per_block=[4, 4], visualize=False,transform_sqrt=True)

#PCA 降维：
    pca = PCA()
    features = pca.fit_transform(features)

#LBP 特征提取：
def texture_detect(image):
    radius = 1
    n_point = radius * 8
    lbp = local_binary_pattern(image, n_point, radius, method='default')
    max_bins = int(lbp.max() + 1)
    hist, _ = np.histogram(lbp, bins=max_bins, range=(0, max_bins))
    return hist

#SIFT 特征提取：
# 定义 FLANN 匹配器函数
def get_flann_matcher():
    flann_params = dict(algorithm=1, trees=5)
    return cv2.FlannBasedMatcher(flann_params, {})



def get_bow_extractor(extract, match):
    return cv2.BOWImgDescriptorExtractor(extract, match)

# 创建 SIFT 特征检测器
def get_extract_detect():
    return cv2.xfeatures2d.SIFT_create(), cv2.xfeatures2d.SIFT_create()

def extract_sift(im, extractor, detector):
    return extractor.compute(im, detector.detect(im))[1]

def extract_sift_point(im, extractor, detector):
    return cv2.drawKeypoints(im, extractor.compute(im, detector.detect(im))[0],
im)

# 创建 BOW 训练器
def bow_features(img, extractor_bow, detector):
    return extractor_bow.compute(img, detector.detect(img))
```

```
def getBowAndDetect(image_list):
    detect, extract = get_extract_detect()
    matcher = get_flann_matcher()
    print("building BOWKMeansTrainer...")
    bow_kmeans_trainer = cv2.BOWKMeansTrainer(1000)

    print("adding features to trainer")
    for image in image_list:
        bow_kmeans_trainer.add(extract_sift(image, detect, extract))
    vocabulary = bow_kmeans_trainer.cluster()
    extract_bow = cv2.BOWImgDescriptorExtractor(extract, matcher)
    extract_bow.setVocabulary(vocabulary)
    return extract_bow, detect

# 训练和测试
    clf = sklearn.svm.SVC(kernel='linear', gamma = 'scale')
    clf.fit(features, labels)
    for feat_path in glob.glob(os.path.join(test_feat_path, '*.feat')):
        total += 1
        if platform.system() == 'Windows':
            symbol = '\\'
        else:
            symbol = '/'
        image_name = feat_path.split(symbol)[1].split('.feat')[0]
        data_test = joblib.load(feat_path)
        data_test_feat = data_test[:-1].reshape((1, -1)).astype(np.float64)
        result = clf.predict(data_test_feat)
        result_list.append(image_name + ' ' + label_map[int(result[0])] + '\n')
        y_true.append(label_map[int(data_test[-1])])
        Y_true.append(int(data_test[-1]))
        y_predict.append(label_map[int(result[-1])])
        Y_pred.append(int(result[-1]))
        if int(result[0]) == int(data_test[-1]):
            correct_number += 1
    print(accuracy_score(Y_true, Y_pred))
    print(precision_score(Y_true, Y_pred))
    print(recall_score(Y_true, Y_pred))
    print(f1_score(Y_true, Y_pred))
    t1 = time.time()
    rate = float(correct_number) / total
    print('准确率是: %f' % rate)
    print('耗时是 : %f' % (t1 - t0))
    return rate
```