利用改进的 LZC 算法对图像与文字进行分类①

曾东海[1], 杨叶芬[2]

[1](广东科学技术职业学院, 广州 510640)
[2](广东科学技术职业学院, 珠海 519090)

摘 要: 在传统二值化方法的基础上, 将信号序列分为多个区间, 进行多尺度二值化粗粒化处理, 不增加符号数目, 得到更精细, 更准确的复杂性. 经过实验, 分析可知文字与图像随着计算尺度的减小, 文字复杂性增大的幅度远小于图像. 通过该结论, 不同尺度下复杂性的差异可以作为图像与文字分类的特征. 因此, 我们可以将改进后的 LZC 算法应用于图像与文字的区分.

关键词: 多尺度; 二值化; 复杂度; LZC; 图像与文字的区分

Classifying the Image and Text by Using the Improved LZC Algorithm

ZENG Dong-Hai[1], YANG Ye-Fen[2]

[1](Guangdong Vocational Institute of Science and Technology, Guangzhou 510640, China)
[2](Guangdong Vocational Institute of Science and Technology, Zhuhai 519090, China)

Abstract: Based on the traditional method of binaryzation, this paper divides the signal sequenceinto into several areas anddomulti-scale binaryzation process on the sequence. Under the condition of without increasing the number of symbols, it can get more meticulous and more accuracy complexity. After the experiment, according to our analysis, with the decrease of the calculating scale, the amplitude of the text's complexity increasing is much smaller than the image's. Acording the conclusion, the difference between the Lempel Ziv complexitys of different scales can be regarded as the feature of image and text. Therefore, it can apply the improved LZC algorithm to distinguish the image and text.

Key words: multi-scale; binaryzation; complexity; Lempel-Ziv Complexity; distinguish image and text

从非线性的复杂信号中提取有效信息是一件不易的事情, 因而人们不得不使用各种方法来提取信号的特征量. 而 Lempel 和 Ziv 提出的 LZC 算法是度量信号复杂程度的简单而快速的算法[1]. 现在 LZC 算法被广泛运用到非线性科学领域的研究中[2,3], 用于提取复杂信号的特征[4], 例如,脑电情感特征提取方面[5], 脑电图 EEG 测量[6,7], 解决了癫痫发作预报计算速度慢的问题[8], 往复压缩机气阀故障诊断[9], 通过对 LZC 算法的改进解决了在度量 HRV 时的噪声干扰问题. 目前用于文字与非文字分类的特征有颜色值的平均值和标准偏差[10], 直方图[11], 最大最小亮度值[12], 纹理特征[13], 投影特征[14], 图像信息度量的图像特征[15], 底层图像特征组合[16]. 从大量文档图像中自动找到我们所需的内容正是从复杂信号中提取特征的一种, LZC 算法自然也可以用于此方面. 但是 LZC 算法的粗粒化处理方法存在缺陷, 如果直接应用可能引起严重的后果, 本文将针对其缺陷对其进行改进, 并将改进后的方法应用于图像与文字的区分.

1 LemPel-Ziv Complexity算法

1976 年, Lempel 和 Ziv 提出了一种复杂度算法, 用于度量随着时间序列的增加, 新模式增加的速度. 我们称这种复杂度为"Lempel-Ziv 复杂度", 其具体算法简述如下:

1) 时间序列的重构: 首先求得目标序列的平均值, 再把这个序列重构成一个符号序列, 令大于平均值的

① 基金项目:广东省自然科学基金(S2013010012920);广东省高等职业教育教学改革项目(201401099)
收稿时间:2016-01-21;收到修改稿时间:2016-04-27 [doi: 10.15888/j.cnki.csa.005486]

x 为“1”, 小于等于平均值的 *x* 为“0”;

2) 对这样的(‘0’, ‘1’)序列中已有的一个子串, 后面再加一个字符 *Q* 或一个字符串 *Q*, 得到一个新的符号序列 *S*, 令 *Sv* 是这一字符串 *S* 减去最后的一个字符, 再看 *Q* 是否属于 *Sv* 字符串中已出现过的子串, 如果属于, 那么把 *Q* 加在后面称之为“复制”, 这时, 可以把 *Q* 延长, 就是增加 *k*, 然后重复上面步骤, 直到 *Q* 不在 *Sv* 中出现过为止; 如果没有出现过称之为“插入”, “插入”时用一个“.”记号放在 *Q* 后; 然后把“.”前面的所有字符看成目标序列, 再重复以上步骤直到序列结束;

3) 第二步完成后, 可以得到一个用“.”分成段的字符串, 段的总数定义为“复杂度”*c(n)*;

4) 根据 Lempel 和 Ziv 的研究, 对几乎所有的 *s*∈*{‘0’, ‘1’}*的 *c(n)*都会按概率唯一趋向一个定值:

$$\lim_{n\to\infty} c(n)=b(n)=\frac{n}{\log_2 n} \tag{1}$$

所以 *b(n)*是序列的渐近行为, 我们可以用它来使 *c(n)*归一化, 成为相对的“复杂性测度”.

$$C_{lz}=\frac{c(n)}{b(n)} \qquad 0\le C_{lz}\le 1 \tag{2}$$

从上述描述可以看出: LemPel-Ziv Complexity(LZC)算法能够反映序列的无序程度. 因此, 基于动力学的 LZC 算法经常被采用来作为一种衡量方法. 例如: (1)LZC 可用于计算不同状态下的 EEG(electroencephalogram, 脑电图)信号, 通过相对的差异, 来区分不同的状态, 并且可以据此对大脑的某些机制作进一步的研究; (2)LZC 可用来区分文本与图像, 其具体方法是本文将着重描述的内容.

2 LZC算法的改进

通过 LZC 算法的简介, 其特点一目了然. 优点: 1)计算 Lempel-Ziv 复杂度需要的数据较短, 计算量不大; 2)无需机器学习理论, 就能提取数据的相关特征及规律. 但是计算 Lempel-Ziv 复杂度的第一步就是将原始时间序列重构为(0, 1)序列, 这种粗粒化处理存在丢失信息, 甚至完全改变原信号动力学特性的危险. 因此, 本文对 LZC 算法进行了改进, 改进 LZC 算法可以从两个方面着手: 1)对信号进行预处理; 2)粗粒化处理方法的改进. 本文主要对粗粒化处理方法的改进进行研究.

对信号的预处理方面, 可以采用小波分解算法对信号的低频部分进行分解, 也可以采用小波包分解算法对高低频都进行分解, 过滤奇异值, 去除噪声. 因为小波分解算法以及小波包分解算法的应用现在已经比较成熟, 本文对该部分不再进行赘述.

2.1 传统粗粒化处理方法

计算 Lempel-Ziv 复杂度的前提和关键是对原始信号进行粗粒化处理, 而传统的粗粒化处理方法就是对原始信号进行二值化处理. 假设已知的时间序列为 *{x(i)|i=1,2,...,n}*, 采用二值化方法对 *x(i)*进行重构, 令

$$x_{ave}=(\sum_{i=1}^{n} x(i))/n \tag{3}$$

式 (3) 中 x_{ave} 原序列 *x(i)* 的平均值, 用 *{S(i)|i=1,2,...,n}*记 *x(i)*重构后的符号序列, 当原序列中的元素 *x(j)*< x_{ave} 时, *S(j)*赋符号 0, 否则赋符号 1, 当 *j* 取遍 1,2,...,n 时, 由此建立一个 0、1 的符号序列 *S(i)*.

2.2 多尺度二值化粗粒化方法

多尺度二值化粗粒化方法是传统粗粒化处理方法的改进, 在传统粗粒化处理方法的基础上, 通过多尺度方法的处理可得到更可靠的序列. 其具体方法如下:

1) 将原始信号分为多个区间, 这个过程相当于粗粒化的过程, 粗粒化的程度由所分区间的数目决定, 区间数目越多, 粗粒化程度越低. 本文用均分法将信号分为 4 个区间: 即先计算出整个信号序列的平均值, 再以平均值为界, 将其分为两个区间, 然后分别对两个区间内的部分求平均值, 这时已划分的两区间又分别被划分成两个区间, 最终使整个信号分成 4 个区间.

2) 如果信号序列的第一个点的值比平均值大, 则该点记为 1, 反之则为 0. 首点的取值方法与传统二值法相同.

3) 对于信号序列中的第二个点以及之后各点, 二值化的结果由其和前一个点的比较决定. 如果增大到了另一个区间, 则该点二值化的值记为 1; 如果减小到了另一个区间, 则该点二值化的值记为 0; 如果处于同一区间内部, 那么该点二值化的取值与前一点一致.

图 1 为例, 图 a 为传统方法将序列分为 2 个区间, 用传统二值化方法重构得到的序列为(11110000). 图 b, 则通过多尺度方法的处理, 把序列分成 4 个区间, 得到重构后的序列为(11010110). 根据传统二值化方法, A、B、C、D 各点都比平均值大, 所以都为 1; E、F、G、H 都小于平均值, 所以都为 0. 而根据多尺度二值化方法, A 点为第一个点, 值大于平均值, 记为 1; B 点

上升到了另一个区间, 为 1; C 点下降到了另一个区间, 为 0; 以后各点依次类推.

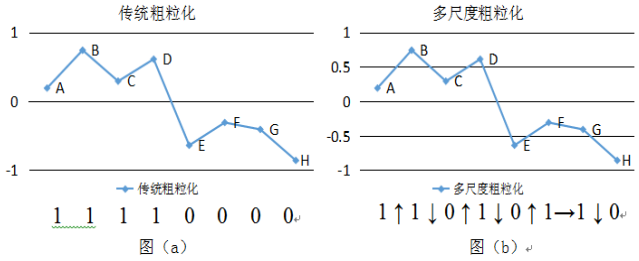


图 1 两种二值化方法比较图

传统的粗粒化处理方法只是表示信号序列大于或者小于均值, 而多尺度二值化粗粒化方法不仅用“0”和“1”更加准确地表述了信号序列的变化, 且“1”表示了信号序列一定尺度的上升趋势以及其延续; “0”表示信号序列一定尺度的下降趋势以及其延续. 比较可知, 多尺度二值化粗粒化方法刻画了一定程度范围内的变化过程, 其精细程度、逻辑性优于传统二值化方法的单纯比较结果.

3 利用LZC算法对图像与文字进行分类

LZC 算法用于分析一维序列, 而图像与文字都是二维图形, 表面上看来, 两者之间是没有联系的. 但是图像与文字可以按照行或者列的方式扫描成一维序列, 这时, 就可以用 LZC 算法对其进行分析, 因此, Lempel-Ziv 复杂度也可以应用到二维图像中. 按照以往经验, 在计算 Lempel-Ziv 复杂度时, 序列长度为 2000 比较合适. 所以, 我们选择图像中 50×50 的正方形区域, 然后将该区域图像变成一维序列, 分析图像与文字在不同尺度下复杂度的差异, 实验结果显示图像和文字的复杂度随着计算尺度的减小都有所增大. 但是图像的复杂度随着计算尺度减小而增大的幅度较大, 文字的复杂度随计算尺度减小而增大的幅度较小. 利用这一差别, 我们可以区分图像与文字.

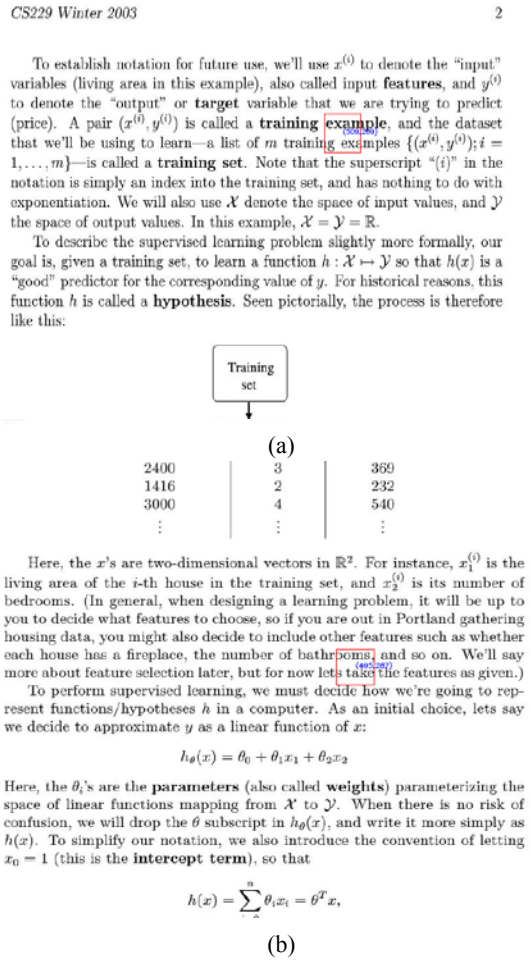
利用LZC算法区分图像与文字的具体方法是在由大到小四个尺度 S1、S2、S3、S4(序列被分为 2、4、8、16 个区间)下分别计算图像的复杂度, 得到结果 LZ1、LZ2、LZ3、LZ4, 利用最小二乘法得到复杂度随尺度变化而变化的斜率 k, 利用斜率 k 的不同区分文字与图像. 在具体计算中, 对于尺寸较小的图像, 只选择 50×50 的区域进行计算; 对于尺寸较大的图像, 根据图像尺寸的不同选择多个区域进行计算, 将几个区域所求得 k 值得均值作为图像的特征.

4 实验

本文做了以下实验分析:

实验样本为纯文字和纯图像各 50 幅, 其中文字中中文、英文各 25 幅, 图像的大小最小的尺寸为 50*50, 最大的为 874*585. 所有图片均以灰度图像进行处理. 进行四个尺度 S1、S2、S3、S4(序列被分为 2、4、8、16 个区间)下的图像的复杂度计算.

图 2 为部分实验样本图, 图 a、b、c、d 为文字图片样本, 图 e、f、g、h 为图像图片样本. 图中红色正方形区域为随机挑选的图像复杂度计算区域. 图 3 为对应的不同尺度下的复杂度的计算结果, 图中蓝色线对应的是纯文字图片的 4 组复杂度折线, 图中红色线对应的是纯图像图片的 4 组复杂度折线. 由图可以看出图像和文字的复杂度随着计算尺度的减小都有所增大. 但是图像的复杂度随着计算尺度减小而增大的幅度较大, 文字的复杂度随计算尺度减小而增大的幅度较小.


CS229 Winter 2003 2

To establish notation for future use, we'll use $x^{(i)}$ to denote the "input" variables (living area in this example), also called input **features**, and $y^{(i)}$ to denote the "output" or **target** variable that we are trying to predict (price). A pair $(x^{(i)}, y^{(i)})$ is called a **training example**, and the dataset that we'll be using to learn—a list of m training examples $\{(x^{(i)}, y^{(i)}); i = 1, \ldots, m\}$—is called a **training set**. Note that the superscript "(i)" in the notation is simply an index into the training set, and has nothing to do with exponentiation. We will also use $\mathcal{X}$ denote the space of input values, and $\mathcal{Y}$ the space of output values. In this example, $\mathcal{X} = \mathcal{Y} = \mathbb{R}$.

To describe the supervised learning problem slightly more formally, our goal is, given a training set, to learn a function $h : \mathcal{X} \mapsto \mathcal{Y}$ so that $h(x)$ is a "good" predictor for the corresponding value of y. For historical reasons, this function h is called a **hypothesis**. Seen pictorially, the process is therefore like this:

Training set

(a)

2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮

Here, the x's are two-dimensional vectors in $\mathbb{R}^2$. For instance, $x_1^{(i)}$ is the living area of the i-th house in the training set, and $x_2^{(i)}$ is its number of bedrooms. (In general, when designing a learning problem, it will be up to you to decide what features to choose, so if you are out in Portland gathering housing data, you might also decide to include other features such as whether each house has a fireplace, the number of bathrooms, and so on. We'll say more about feature selection later, but for now lets take the features as given.)

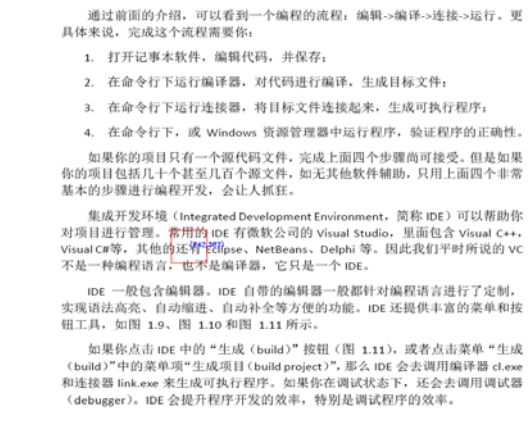
To perform supervised learning, we must decide how we're going to represent functions/hypotheses h in a computer. As an initial choice, lets say we decide to approximate y as a linear function of x:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Here, the θ_i's are the **parameters** (also called **weights**) parameterizing the space of linear functions mapping from $\mathcal{X}$ to $\mathcal{Y}$. When there is no risk of confusion, we will drop the θ subscript in $h_\theta(x)$, and write it more simply as $h(x)$. To simplify our notation, we also introduce the convention of letting $x_0 = 1$ (this is the **intercept term**), so that

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

(b)


通过前面的介绍，可以看到一个编程的流程：编辑->编译->连接->运行。更具体来说，完成这个流程需要你：

1. 打开记事本软件，编辑代码，并保存；
2. 在命令行下运行编译器，对代码进行编译，生成目标文件；
3. 在命令行下运行连接器，将目标文件连接起来，生成可执行程序；
4. 在命令行下，或 Windows 资源管理器中运行程序，验证程序的正确性。

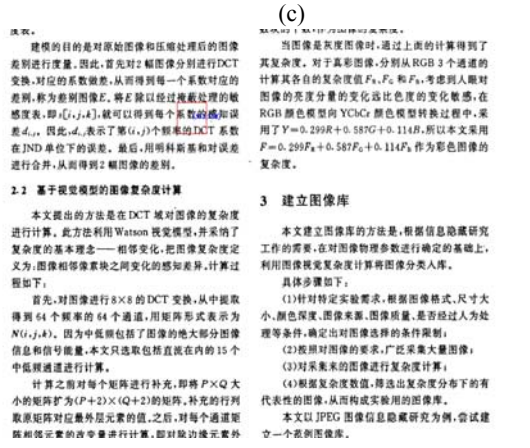
如果你的项目只有一个源代码文件，完成上面四个步骤尚可接受。但是如果你的项目包括几十个甚至几百个源文件，如无其他软件辅助，只用上面四个非常基本的步骤进行编程开发，会让人抓狂。

集成开发环境（Integrated Development Environment，简称 IDE）可以帮助你对项目进行管理。常用的 IDE 有微软公司的 Visual Studio，里面包含 Visual C++，Visual C#等，其他的还有 Eclipse、NetBeans、Delphi 等。因此我们平时所说的 VC 不是一种编程语言，也不是编译器，它只是一个 IDE。

IDE 一般包含编辑器。IDE 自带的编辑器一般都针对编程语言进行了定制，实现语法高亮、自动缩进、自动补全等方便的功能。IDE 还提供丰富的菜单和按钮工具，如图 1.9、图 1.10 和图 1.11 所示。

如果你点击 IDE 中的“生成（build）”按钮（图 1.11），或者点击菜单“生成（build）”中的菜单项“生成项目（build project）”，那么 IDE 会去调用编译器 cl.exe 和连接器 link.exe 来生成可执行程序。如果你在调试状态下，还会去调用调试器（debugger）。IDE 会提升程序开发的效率，特别是调试程序的效率。

(c)


建模的目的是对原始图像和压缩处理后的图像差别进行度量。因此，首先对2幅图像分别进行DCT变换，对应的系数做差，从而得到每一个系数对应的差别，称为差别图像E。将E除以经过掩蔽处理的敏感度表，即$e[i,j,k]$，就可以得到每个系数的感知误差$d_{i,j,k}$。因此，$d_{i,j}$表示了第(i,j)个频率的DCT系数在JND单位下的误差。最后，用明科斯基和对误差进行合并，从而得到2幅图像的差别。

2.2 基于视觉模型的图像复杂度计算

本文提出的方法是在DCT域对图像的复杂度进行计算。此方法利用Watson视觉模型，并采纳了复杂度的基本理念——相邻变化，把图像复杂度定义为：图像相邻像素块之间变化的感知差异，计算过程如下：

首先，对图像进行8×8的DCT变换，从中提取得到64个频率的64个通道，用矩阵形式表示为$N(i,j,k)$。因为中低频包括了图像的绝大部分图像信息和信号能量，本文只选取包括直流在内的15个中低频通道进行计算。

计算之前对每个矩阵进行补充，即将$P\times Q$大小的矩阵扩为$(P+2)\times(Q+2)$的矩阵，补充的行列取原矩阵对应最外层元素的值，之后，对每个通道矩阵相邻元素的差值进行计算，即对除边缘元素外

当图像是灰度图像时，通过上面的计算得到了其复杂度。对于真彩图像，分别从RGB 3个通道的计算其各自的复杂度值F_R、F_G和F_B，考虑到人眼对图像的亮度分量的变化远比色度的变化敏感，在RGB颜色模型向YCbCr颜色模型转换过程中，采用了$Y=0.299R+0.587G+0.114B$，所以本文采用$F=0.299F_R+0.587F_G+0.114F_B$作为彩色图像的复杂度。

3 建立图像库

本文建立图像库的方法是，根据信息隐藏研究工作的需要，在对图像物理参数进行确定的基础上，利用图像视觉复杂度计算将图像分类入库。

具体步骤如下：

(1)针对特定实验需求，根据图像格式、尺寸大小、颜色深度、图像来源、图像质量、是否经过人为处理等条件，确定出对图像选择的条件限制；

(2)按照对图像的要求，广泛采集大量图像；

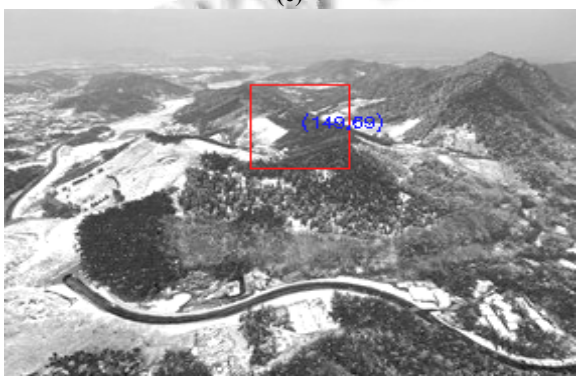
(3)对采集来的图像进行复杂度计算；

(4)根据复杂度数值，筛选出复杂度分布下的有代表性的图像，从而构成实验用的图像库。

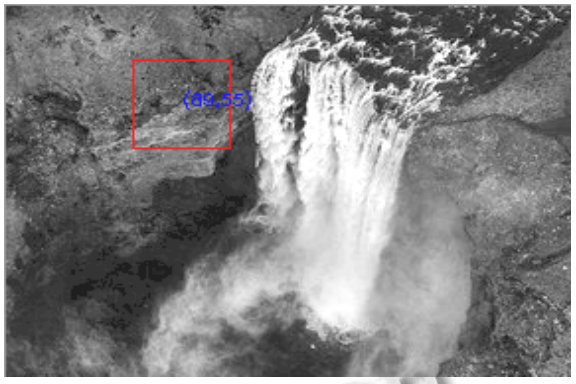
本文以JPEG图像信息隐藏研究为例，尝试建立一个盗图图像库。

(d)

(e)



(f)



(g)

(h)

图 2　部分样本图片及复杂度计算区域

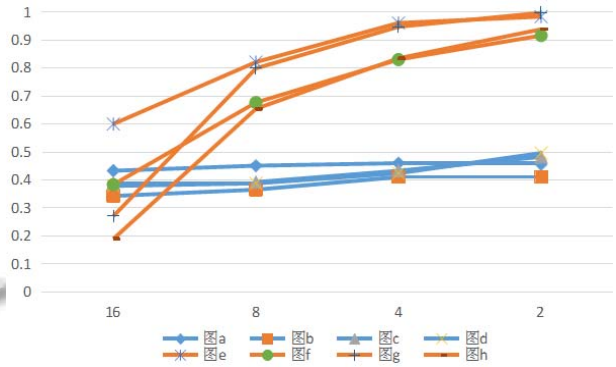


图 3　部分样本在不同尺度下复杂度计算结果

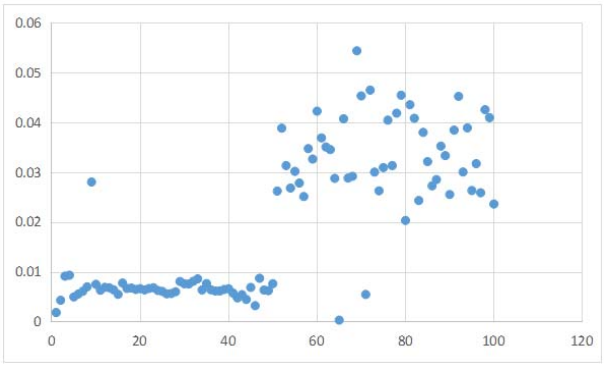


图 4　所有实验样本的斜率均值 k 的分布图

图 4 为所有实验样本的斜率 k 的均值, 前 50 组为纯文字图片, 后 50 组为纯图像图片, 由曲线发现在文字组合图像组中均出现异常值, 总结出现情况如下:

① 对于文字而言, 如果图片噪声干扰较严重的, 其复杂度计算也将受到影响, 因此可以在计算复杂度之前, 对图片进行适当的滤波去噪处理, 如图 5(a).

② 图像为二值化后的图片, 如图 5(b), 对于二值化图片的干扰将严重影响文字与图像的区分, 如果输入为二值化图像, 随着尺度变化, 复杂度的变化几乎为 0.

③ 对于图像而言, 如果图像中的背景灰度值非常单一的话, 也将影响到复杂度的计算, 将随着尺度的减小而变化不大,如图 5(c).

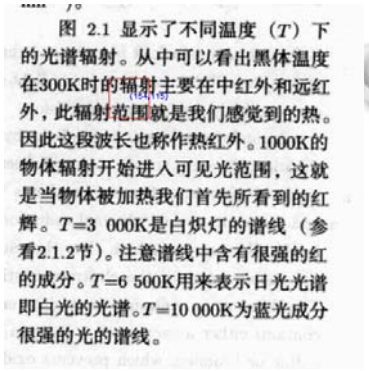
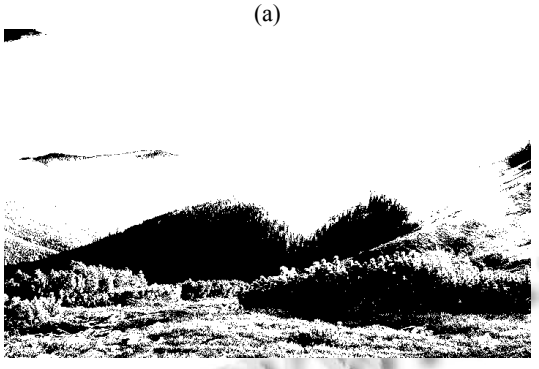


图 2.1 显示了不同温度（T）下的光谱辐射。从中可以看出黑体温度在300K时的辐射主要在中红外和远红外，此辐射范围就是我们感觉到的热。因此这段波长也称作热红外。1000K的物体辐射开始进入可见光范围，这就是当物体被加热我们首先所看到的红辉。T=3 000K是白炽灯的谱线（参看2.1.2节）。注意谱线中含有很强的红的成分。T=6 500K用来表示日光光谱即白光的光谱。T=10 000K为蓝光成分很强的光的谱线。

(a)



(b)

(c)

图 5 异常图像

排除了以上这些干扰因素的影响之后, 我们可以发现其中前 50 组为文字的斜率均值 k 小于 0.01, 后 50 组为图像的斜率均值 k 大于 0.02. 由此我们可以利用 k 值为 0.015 作为区分文字和图像的阈值.

以下为测试效果, 选用 50*50 的正方形框作为扫描区域, 自左向右, 自上而下扫描整幅测试图片, 如果该区域的 k 值大于阈值, 那么保留原图, 否则涂黑, 效果如图 6.

图 6 测试原图及搜索结果

5 结语

本文针对 LZC 算法的缺陷, 做出了相应的改进, 并将该改进后的算法, 应用于图像与文字的区分. 该方法不需要增加符号数目, 计算方法容易实现, 且精细度、准确度较好, 可以较好地区分图像与文字. 后续我们将会从以下几方面进行进一步的研究: 1)完善输入图像的去噪方法; 2)如何将复合图像和单一内容的图像区分开; 3)将不同尺度下的复杂性方法作为一个特征, 和图像的其它特征相结合, 利用现有分类工具对图像进行分类, 各分类特征之间互相弥补不足,提高分类精度, 增加实用性.

参考文献

1 DianZhong Z. Research on the correlation between the mutual information and Lempel-Ziv complexity of nonlinear time series. Acta Phys. Sin., 2007, 56(6): 3152–3157.

2 张佃中,谭小红,王智,刘昭前.基于等概率粗粒化的复杂度算法及其应用.系统仿真学报,2008,20(15):4096–4098.

3 解幸幸,李舒,张春利,李建康.Lempel-Ziv 复杂度在非线性检测中的应用研究.复杂系统与复杂性科学,2005,2(3):61–66.

4 Kaspar F, Schuster HG. Easily calculable measure for the complexity of spatiotemporal patterns. Physical Review A, 1987, 36(2): 842–848.

5 张栋.Lempel-Ziv 复杂度的尺度划分方法的研究及应用[硕士学位论文].太原:太原理工大学,2013.

6 Ibanez-Molina AJ, Iglesias-Parro S, Soriano MF, Aznarte JI. Multiscale Lempel-Ziv complexity for EEG measures. Clinical Neurophysiology Official Journal of the International Federation of Clinical Neurophysiology, 2015, 126(3): 541–548.

7 Fernandez A, Lopez-Ibor MI, Turrero A, Santos JM, Moron MD, Hornero R, Gomez C, Mendez MA, Ortiz T, Lopez-Ibor JJ. Lempel-Ziv complexity in schizophrenia: A MEG study. Clinical Neurophysiology Official Journal of the International Federation of Clinical Neurophysiology, 2011, 122(11): 2227–2235.

8 韩敏,曹占吉,孙磊磊,洪晓军.基于 AR 模型和 Lempel-Ziv 复杂度的癫痫发作预报.北京生物医学工程,2012,31(3):476–480.

9 唐友福,刘树林,刘颖慧,姜锐红.基于非线性复杂测度的往复压缩机故障诊断.机械工程学报,2012,48(3):102–107.

10 Somporn C, Lursinsap C, Sophasathit P, Siripant S. Fuzzy c-mean: A statistical feature classification of text and image segmentation method. International Journal of Uncertainty Fuzziness and Knowledge-Based Systems, 2001, 9(6): 661–671.

11 Pascovici A, Shu JS. Method and apparatus for processing a document by segmentation into text and image areas. US: US5883973, 1999.

12 Kawanaka S, Ida S, Takemura H. Method for discriminating between figure and text areas of an image. US: US5381241A, 1995.

13 刘仁金,高远飙,郝祥根.文本图像页面分割算法研究.中国科学技术大学学报,2010,40(5):500–504.

14 邱立松,黄继风.文本图像信息的提取与识别.计算机与数字工程,2013,41(12):1981–1984.

15 童莉,平西建.基于信息度量的图像特征与文本图像分类.计算机工程,2004,30(17):143–145.

16 曾东红,黄朝志,黄细妹.基于底层图像特征组合的文本图像分类研究.江西理工大学学报,2013,34(5):82–87.