软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(4):1173−1188 [doi: 10.13328/j.cnki.jos.005674]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

一种自适应在线核密度估计方法*

邓齐林[1,2], 邱天宇[1,2], 申富饶[1,2], 赵金熙[1,2]

[1](计算机软件新技术国家重点实验室(南京大学),江苏 南京 210023)

[2](南京大学 计算机科学与技术系,江苏 南京 210023)

通讯作者: 申富饶, E-mail: frshen@nju.edu.cn

摘 要: 给定一组观察数据,估计其潜在的概率密度函数是统计学中的一项基本任务,被称为密度估计问题.随着数据收集技术的发展,出现了大量的实时流式数据,其特点是数据量大,数据产生速度快,并且数据的潜在分布也可能随着时间而发生变化,对这类数据分布的估计也成为亟待解决的问题.然而,在传统的密度估计算法中,参数式算法因为有较强的模型假设导致其表达能力有限,非参数式算法虽然具有更好的表达能力,但其计算复杂度通常很高.因此,它们都无法很好地应用于这种流式数据的场景.通过分析基于竞争学习的学习过程,提出了一种在线密度估计算法来完成流式数据上的密度估计任务,并且分析了其与高斯混合模型之间的密切联系.最后,将所提算法与现有的密度估计算法进行对比实验.实验结果表明,与现有的在线密度估计算法相比,所提算法能够取得更好的估计结果,并且能够基本上达到当前最好的离线密度估计算法的估计性能.

关键词: 密度估计;高斯混合模型;数据流;在线学习;竞争学习

中图法分类号: TP181

中文引用格式: 邓齐林,邱天宇,申富饶,赵金熙.一种自适应在线核密度估计方法.软件学报,2020,31(4):1173−1188. http://www.jos.org.cn/1000-9825/5674.htm

英文引用格式: Deng QL, Qiu TY, Shen FR, Zhao JX. Adaptive online kernel density estimation method. Ruan Jian Xue Bao/ Journal of Software, 2020,31(4):1173−1188 (in Chinese). http://www.jos.org.cn/1000-9825/5674.htm

Adaptive Online Kernel Density Estimation Method

DENG Qi-Lin[1,2], QIU Tian-Yu[1,2], SHEN Fu-Rao[1,2], ZHAO Jin-Xi[1,2]

[1](State Key Laboratory for Novel Software Technology (Nanjing University), Nanjing 210023, China)

[2](Department of Computer Science and Technology, Nanjing University, Nanjing 210023, China)

Abstract: Based on observed data, density estimation is the construction of an estimate of an unobservable underlying probability density function. With the development of data collection technology, real-time streaming data becomes the main subject of many related tasks. It has the properties of that high throughput, high generation speed, and the underlying distribution of data may change over time. However, for the traditional density estimation algorithms, parametric methods make unrealistic assumptions on the estimated density function while non-parametric ones suffer from the unacceptable time and space complexity. Therefore, neither parametric nor non-parametric ones could scale well to meet the requirements of streaming data environment. In this study, based on the analysis of the learning strategy in competitive learning, it is proposed a novel online density estimation algorithm to accomplish the task of density estimation for such streaming data. And it is also pointed out that it has pretty close relationship with the Gaussian mixture model. Finally, the proposed algorithm is compared with the existing density estimation algorithms. The experimental results show that the proposed

* 基金项目: 国家自然科学基金(61876076); 江苏省自然科学基金(BK20171344)

Foundation item: National Natural Science Foundation of China (61876076); Natural Science Foundation of Jiangsu Province of China (BK20171344)

收稿时间: 2017-03-03; 修改时间: 2018-04-02; 采用时间: 2018-09-17; jos 在线出版时间: 2019-05-22

CNKI 网络优先出版: 2019-05-22 15:54:33, http://kns.cnki.net/kcms/detail/11.2560.TP.20190522.1554.015.html

algorithm could obtain better estimates compared with the existing online algorithm, and also get comparable estimation performance compared with state-of-the-art offline density estimation algorithms.

Key words: density estimation; Gaussian mixture model; data stream; online learning; competitive learning

在密度估计任务中,给定某个未知分布的代表性样本数据,我们需要从这些数据中尽可能准确地恢复产生其概率密度或者概率分布函数.随机变量的概率密度函数描述了特征空间的数据分布情况,因此,它能够为数据驱动的决策过程提供非常重要的判断依据信息,并经常被用于和数据分析相关的其他领域.例如,在机器学习任务中,为训练数据估计概率分布并不是我们的最终目的,我们最终还是希望能够基于分布信息对未知样本进行可靠的预测,从而帮助我们做出相关的任务决策.从生成式分类模型的角度来看,首先需要估计出生成样本数据的类别条件概率密度,然后结合贝叶斯推理原则,就能得到该样本数据属于各个类别的条件概率,继而做出分类决策.与判别式模型相比,通过对生成样本的概率密度函数进行建模,我们能够对样本数据的产生和分布有更加深刻的理解和认识,并有助于从理论上来分析模型的学习性能.

密度估计作为一种通用的数据分析方法,其本身并不依赖于具体的问题.所以,相对于许多机器学习的具体任务而言,密度估计过程更具有普遍性,并且能够结合成熟的统计推理方法,从而保证了理论上的可靠性.这使其成为帮助我们进行数据分析和完成学习任务的有效手段.例如,在异常检测任务中,目前主流的异常检测算法大多是基于对数据的概率密度进行建模[1−3].具体来说,我们可以首先为正常的样本建立概率分布模型,然后将该模型用于测试样本,如果测试样本位于正常概率模型的低密度区域,那么该数据就很有可能是异常数据.这种基于密度的方法能够直观地分析数据的分布情况,从而帮助研究人员做出最佳判断.在计算机视觉领域,也有很多密度估计应用的地方[4,5].例如,通过分析图片中的密度信息来对背景进行建模就用到了密度估计算法[6−8].此外,数据的分布曲线通常是人们观察数据分布情况的最直接的途径,密度估计还能够为数据可视化提供支持[9,10],从而帮助我们更好地了解数据的内在特性.

随着现在收集到的可用数据越来越多,如何从分布复杂多变的大量数据中获取有用的信息成为机器学习领域越来越重要的问题.例如:股票高频交易过程中的快速分析、实时传感器数据的处理、搜索引擎海量文档的分类等等.事实上,在大规模数据场景下,即使是最简单的机器学习算法可能也面临着巨大的计算挑战,传统的基于离线训练的算法很多时候已经不能满足实际需求.类似地,基于密度估计的学习算法在这种场景下的计算复杂度通常很高,甚至难以进行.因此,我们需要设计新的算法使其能够适用于这些大规模数据集上的数据分析和学习任务.面对这种计算困难的学习问题,在线学习为我们提供了一种可行的学习策略,与离线学习方式不同,数据是依次输入到在线学习模型的,每次接收到新数据后,学习系统使用新数据来更新当前模型以得到一个更好的模型.在本文中,为尝试解决大规模数据集上的学习任务,提出了一种在线核密度估计方法.该方法将在线学习的思想引入到了传统的密度估计问题上,使其具有在线学习数据概率分布密度的能力.此外,实际应用场景中的数据分布可能随着时间发生变化,而在线学习也能够处理这种分布不断变化的非稳态数据.

本文第 1 节介绍一些密度估计的相关工作.第 2 节介绍我们提出的在线密度估计算法.第 3 节对我们的方法进行更加细致的分析.第 4 节给出我们的算法与现有的密度估计算法的对比实验结果,并进行比较分析.第 5 节对全文进行总结.

1 相关工作

密度估计问题可以描述为:假设随机变量 $X \in R^d$ 服从一个未知的概率密度函数 $f(x)$,给定一组 X 的随机样本 $x_1,x_2,\ldots,x_n \in R^d$, 需要构造一个 $f(x)$的估计 $\hat{f}(x)$, 使其尽可能地接近真实分布 $f(x)$.现在已有的密度估计算法按照模型假设大致可以被分为参数式(parametric)方法和非参数式(nonparametric)方法.

描述性统计学的研究表明,存在多种形式的分布函数能够描述自然界中的数据和现象.例如泊松分布能够描述单位时间内罕见事件的发生次数,而高斯分布能够描述测量误差.因此,对于这类密度估计问题,一个自然的想法就是先假设其服从某个分布形式,只是分布函数的参数未知,于是估计概率密度函数也就是估计其中的

未知参数.这种方法被称为参数式方法,因为我们对待估计的密度函数作了形式上的假设(比如高斯分布).对于参数式方法,通常使用最大似然或者最大后验的原则进行参数估计,其代表方法是高斯混合模型(Gaussian mixture model,简称 GMM)[11].GMM 是一种特殊的有限混合模型(finite mixture model)[11,12],它使用一组加权的高斯分布来近似目标概率分布:

$$f(x)=\sum_{k=1}^{K}\alpha_k\phi(x|\mu_k,\Sigma_k) \tag{1}$$

其中,K 是高斯分量的个数,α_k 是各个分量所占的权重,$\phi(x)$ 为高斯分布的概率密度函数.GMM 通常使用 EM 算法来进行学习,由于目标概率密度函数的非凸性,EM 算法只能保证收敛至局部最优点而非全局最优点.此外,训练过程中 K 作为一个超参数,决定了该模型对数据分布的拟合能力,通常需要反复运行学习算法,通过比较各项性能指标来确定合适的取值.

与参数式方法不同,非参数方法对于密度函数不作任何形式上的假设,这类方法通常依赖于统计理论中的渐进性质来构造一组逐渐逼近要估计的密度函数序列,其代表方法是核密度估计(kernel density estimation,简称 KDE)[13].KDE 使用所有的样本信息来近似目标概率分布:

$$f_h(x)=\frac{1}{n}\sum_{i=1}^{n}K_h(x-x_i) \tag{2}$$

其中,n 表示密度估计使用的样本数量,$K(\cdot)$为核函数(比如高斯核).KDE 没有训练过程,核函数中带宽参数 h 作为一个取值非常关键的超参数,用于控制模型的平滑程度,其取值越大,则得到的概率密度曲线就越平滑.在一维情况下,理论上存在选取 h 的最优值,尤其是在对于实际分布是正态分布时,我们可以直接给出最优的带宽参数.然而在多维情况下,h 的设定并没有一种确定性的方法,通常需要多次比较才能确定合适的取值.

对比密度估计的参数式和非参数式方法,我们可以看到:参数式方法因为假设样本数据来自某一特定的分布,先验地让模型具有了较强的概率分布假设,但实际数据可能并不完全符合假设的分布,因此其在实际应用上受到了限制;而非参数式算法的特点是几乎不需要任何特别的假设就能提供理论上比较完备的密度估计结果,但它们通常受限于计算复杂度,只适合处理低维和小数据样本.针对现有方法的这些问题,研究者们提出了很多改进算法.文献[14]通过引入核函数,将支持向量的思想引入到 KDE 中,将原来的 KDE 问题转化为一个标准的 SVM 问题.RSDE[15]在 KDE 的基础上引入了一个能够得到稀疏解的优化过程,从而大大减少了传统 KDE 中需要保存的样本数量.在最终的估计模型里,只有一些关键的样本被保留下来,因此降低了模型的复杂度,但同时也引入了一个额外的全局二次优化过程.RKDE[16]在 KDE 的形式基础上引入了核函数,将 KDE 解释为样本均值与要估计的数据点的内积,然后通过鲁棒的均值估计方法实现鲁棒的密度估计,该方法适合于带有噪音数据的密度估计问题,并且需要进行离线训练.KDEd(KDE via diffusion,本文简称 KDEd)[17]是一种自适应地选择 KDE 带宽参数的密度估计算法,该算法从线性扩散过程的一些物理性质中得到启发,能够通过一种全局算法自动确定合适的带宽,KDEd 训练时需要利用所有数据,所以也是一种离线式的密度估计算法.

离线密度估计算法需要同时使用全部的数据进行学习,计算复杂度非常高.为了克服这个缺点,人们相继提出了很多在线式更新模型的方法.SOMN[18]使用自组织映射(self-organizing map,简称 SOM)网络作为框架,提出了一种以最小化 KL 距离为优化目标的密度估计算法,其学习过程因为被限制在一组结构上互相靠近的 SOM 神经元上,因此能够加速算法的收敛.然而,因为 SOM 需要预先确定网络的结构,导致 SOMN 神经元的数量以及它们的拓扑结构无法灵活地适应数据分布的变化.文献[19]提出了一种改进高斯混合模型的学习算法,高斯分量的数量不再固定,而是通过一种贪心策略不断增加高斯分量,直到一个预先指定的上限为止.但是,由于该学习算法中存在一组全局搜索过程,因此在学习过程中需要保留所有的训练数据.文献[20]在 KDE 的基础上使用 mean-shift[21]来探测模型中的高密度区域,然后在该位置动态创建或者删除高斯分量来实现在线学习.文献[22]提出的算法能够根据当前模型对数据进行在线密度估计,但是,其中为了降低模型复杂度而使用的合并算法借助了统计检验的结果,该检验方式每次都需要一组数量足够多的样本才能够得到有效的结果.IGMM[23]是一种完全增量式的高斯混合模型,不需要保留原始训练数据,模型的每次更新只依赖于当前的输入

数据.但是该学习算法总是会全局性地更新模型,这种做法不利于学习复杂的局部分布,而且收敛速度较慢.oKDE[24]把混合模型和 KDE 结合起来实现在线密度估计,它在学习过程中维护一个当前已经学习过数据的混合模型,然后使用 KDE 方法从当前输入数据和混合模型中得到当前的密度估计结果,最终估计的概率密度函数被定义为对当前模型进行卷积的结果.然而这种卷积是一种整体性的平滑操作,因此,oKDE 往往会使得估计结果过于平滑,同样缺少对局部分布的学习能力.文献[25]中的方法与 oKDE 类似,但却引入了负样本对数据进行交互式学习以提高估计的精确度,并使用一种压缩算法来降低模型的复杂度.文献[12]提出了一种基于增量高斯混合模型的在线密度估计方法

综上所述,目前对于大规模数据集上的密度估计任务还没有非常有效的算法,一方面是因为现有的很多密度估计算法都是基于离线算法进行训练,也就是训练时需要获取到所有的训练数据;另一方面,现有的在线算法缺少局部学习的能力,因此容易破坏已经学习到的有效分布信息.针对以上介绍的密度估计问题以及现有方法的缺陷,本文将在线学习的思想与传统的密度估计问题相结合,提出一种新的在线密度估计算法,该算法不需要指定关于训练数据的分布结构等先验信息,而是在学习过程中自适应地调整模型以适应输入数据分布的变化.

2 自适应在线核密度估计

本文针对大规模数据的密度估计问题,提出了一种自适应的在线核密度估计(adaptive online kernel density estimation)算法,该算法通过最大化每个局部高斯分量周围训练数据的似然值自适应地对模型进行在线更新.它可以看作是一种增量式的高斯混合模型,其中使用了一个特殊的阈值参数来控制高斯分量的生成.同时,高斯分量的个数能够随着数据分布的变化而自适应地增加或者减少.此外,我们还使用了一个局部更新策略,能够使用每个数据样本对其周围的高斯分量进行在线学习.这使得我们的模型在降低训练复杂度的同时能够发现局部变化较为复杂的密度分布.图 1 显示了算法的总体框架,主要包括 3 个部分.

(1) 激活高斯分量.为每个新输入数据确定需要更新的高斯分量集合,或者增加新的高斯分量;

(2) 更新高斯分量.确定了需要更新的高斯分量后,用最大似然方法更新这些分量的参数;

(3) 估计器去燥.删除密度估计器中那些很可能是由孤立的噪音数据而错误生成的高斯分量.

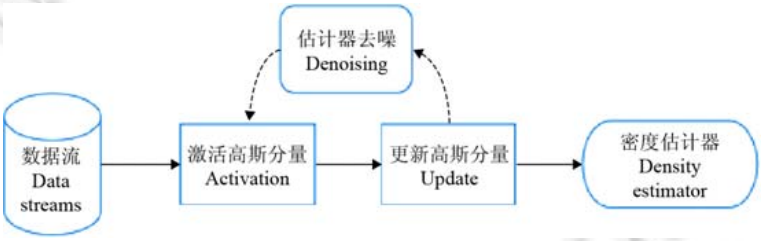

Fig.1 Framework of the proposed method

图 1 提出的算法总体框架

下面分别对这 3 个部分进行详细的介绍,其中涉及到的主要符号说明见表 1.作为一个密度估计模型,我们的算法最终估计的密度函数形式是高斯分布组成的混合分布的形式.但与 GMM 不同,在我们的模型中,高斯分量的数量是由学习算法根据输入数据的分布情况自动确定的.在学习过程中,每一个高斯分量都能够用一个四元组(μ_k,Σ_k,n_k,T_k)来描述,其中,μ_k 是均值向量,Σ_k 是协方差矩阵,n_k 用来衡量该分量对当前所有样本的贡献程度,或者可以认为是数据流中属于该分量的样本数目,T_k 是相似度阈值参数,它控制了该高斯分量对周围输入数据的影响范围.最终,我们的密度估计模型给出的全局概率密度函数形式为

$$p(x)=\sum_{k=1}^{K}w_k\phi(x\mid\mu_k,\Sigma_k) \tag{3}$$

其中,w_k 是高斯分量 k 的混合系数.在我们的模型中,其值为 $w_k=n_k/\sum_{k'}n_{k'}$,它与该分量对整体分布的贡献程度有关,μ_k 和 Σ_k 以及 $\phi(x)$ 的含义与 GMM 的模型参数相同.

Table 1 Symbol description

表 1 符号说明

符号	含义说明
x	输入样本数据
$\phi(x)$	高斯分布的概率密度函数
w_i	高斯分量的权重
$D_i(x)$	样本与高斯分量之间的马氏距离
μ_k	高斯分量的均值
Σ_k	高斯分量的协方差
n_k	高斯分量的权重系数
T_k	高斯分量的相似度阈值

2.1 激活高斯分量

假设当前我们已经学到的模型为 $\{(\mu_k,\Sigma_k,n_k,T_k)|1\leqslant k\leqslant K\}$, 其中,$K$ 是当前模型中高斯分量的个数,其值会随着学习过程的进行动态地发生变化.当输入一个新数据 $x\in R^d$ 时,我们首先计算它与各个高斯分量的马氏距离 $D_k(x)$:

$$D_k(x)=\sqrt{(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)},k=1,2,\ldots,K \tag{4}$$

这里选择马氏距离而不是欧式距离的原因是:通过引入协方差矩阵信息,马氏距离能够更好地描述各向异性的数据分布.然后,我们构造输入数据 x 激活的高斯分量集合 S:

$$S=\{k\,|\,k\in\{1,2,\ldots,K\},D_k(x)<T_k\} \tag{5}$$

根据定义可以看出,集合 S 实际上包含了当前模型中那些与 x 马氏距离小于其阈值 T_i 的高斯分量 k,这些高斯分量生成该样本点的概率是最大的,它们的参数会被更新以适应新输入数据 x.同时,T_i 作为高斯分量 i 的相似度阈值参数,它影响了每个高斯分量对周围训练数据的敏感程度,T_i 越小,则学习算法更倾向于学习局部分布,反之,则倾向于进行全局性的更新.为了增加学习过程的灵活性,可以增加一个用户自定义参数 η,使用阈值 ηT_i.

我们根据输入数据与现有模型之间的相关性得到激活集合 S,而如果输入数据 x 与当前的模型差异性比较大,出现集合 $S=\varnothing$ 的情况,那么就意味着当前模型不能很好地适应新数据 x,所以我们就需新增加一个高斯分量来拟合该数据,这是一种动态适应数据分布变化的策略.新的高斯分量的统计量初始化为

$$\mu_{new}=x,\Sigma_{new}=\sigma I,n_{new}=1,T_{new}=\sqrt{\chi^2_{d,q}} \tag{6}$$

其中,I 为相应维数的单位矩阵,σ是一个用户自定义的参数, $\chi^2_{d,q}$ 是自由度为 d 的 χ^2 分布(chi-square distribution)的 q 分位点,我们知道,χ^2 分布是由标准正态分布构造而成的一个概率分布模型,当其自由度很大时,也近似形成一个正态分布,本文后面部分会对其进行详细分析.

2.2 更新高斯分量

当新输入数据激活了集合 S 中的高斯分量后,如果 $S\neq\varnothing$,我们将模型的参数更新限制为 S 中的那些高斯分量,而不是整个模型中的高斯分量.这种局部学习的策略不仅加速了模型更新时的计算速度,同时也使得模型本身能够在学习新数据时不影响那些距离较远已经学到的分布信息.这种性质对于在线学习是非常有必要的,因为在在线学习背景下,能够获得的数据信息具有局部性,永远不可能获取训练数据的全局信息.只在局部进行更新也是应对非稳态数据的一种安全策略,这样可以把模型误差控制在局部范围内.

为说明如何对集合 S 中的高斯分量进行更新,我们首先考虑单个高斯分布的情形,设 x_n 是数据流中当前的输入样本,μ^n 是当前所有样本的均值,Σ^n 是当前所有样本的协方差.通过分析均值向量和协方差矩阵的最大似然估计结果,很容易得到它们的增量式递推更新方程:

$$\mu^n=\mu^{n-1}+\frac{1}{n}(x_n-\mu^{n-1}) \tag{7}$$

$$\Sigma^n=\frac{n-1}{n}\Sigma^{n-1}+\frac{n-1}{n^2}(x_n-\mu^{n-1})^T(x_n-\mu^{n-1}) \tag{8}$$

把这个更新策略推广到 S 中所有的高斯分量上,使得它们都可以针对新数据更新自身模型参数.为此,我们

首先确定当前输入 x_t 对每个分量 k 的更新程度 r_k^t:

$$r_k^t = \frac{\phi\left(x_t|\theta_k^{t-1}\right)}{\Sigma_{k'}\phi(x_t|\theta_{k'}^{t-1})} \tag{9}$$

这里,我们使用了每个分量生成该新样本数据的概率自适应确定.这种策略的直观意义就是:如果分量 k 生成当前样本 x_t 的概率越大,那么就应该对其更新的越多.于是就有:

$$n_k^t = n_k^{t-1} + r_k^t \tag{10}$$

$$\mu_k^t = \mu_k^{t-1} + \frac{1}{n_k^t}\left(x_t - \mu_k^{t-1}\right) \tag{11}$$

$$\Sigma_k^t = \frac{n_k^t - 1}{n_k^t}\Sigma^{t-1} + \frac{n_k^t - 1}{(n_k^t)^2}\left(x_t - \mu_k^{t-1}\right)^T\left(x_t - \mu_k^{t-1}\right) \tag{12}$$

其中,n_k 用来调整高斯分量的更新步长.这里,n_k 的更新策略意味着对于每个输入数据,其激活的集合 S 中所有高斯分量按照自身的贡献度进行参数更新.注意到,当 S 中存在多个高斯分量时,r_k 使得集合 S 中的每个高斯分量都具有了一定的学习能力,而当 S 中只有一个高斯分量时,这个更新策略则退化为单个高斯分布的情况.

2.3 模型去噪

对于实际应用中的密度估计任务,训练数据中普遍包含许多噪音数据.通常,我们有理由假设这些噪音数据分布在目标概率密度函数取值较低的区域中,而且噪音数据的密度足够低,这样它们才不至于破坏正常数据的分布趋势,导致模型无法学到真实的分布信息.根据上文介绍的增加高斯分量的原则,一个出现在低密度区域的噪音数据很容易引起一次高斯分量的动态生成,但是该分量在接下来的很长一段学习时间内很可能不会出现在输入数据的邻域集合内,因此这些分量就会很少参与模型更新.因此,在我们的密度估计模型中,采取了动态删除由噪音数据产生的高斯分量的策略:周期性地删除那些在学习过程中参与更新程度始终较低的高斯分量.

作为一种启发式准则,我们使用高斯分量吸收的训练样本数来衡量其参与模型更新的程度.具体来说,因为噪音数据生成的高斯分量的有效训练样本数相对于非噪音数据生成的高斯分量来说应该非常小,为此,首先计算当前模型的平均有效训练样本数:

$$\overline{n} = \frac{1}{K}\sum_{i=1}^{K} n_i \tag{13}$$

其中,K 为当前的高斯分量数目.每学习过若干个训练样本(用户自定义参数 λ),就删除那些有效训练样本数较少(也就意味着很少参与模型更新)的分量集合 S_{delete},其定义如下:

$$S_{delete} = \left\{C_i | i \in \{1,2,\ldots,K\}, n_i < v\overline{n}\right\} \tag{14}$$

其中,$v \in (0,1)$ 为用户自定义参数,如果训练数据集中包含的噪音数据比较多,则应该设置较大的,同时可以设置较小的去燥周期.

传统的 GMM 在训练时没有很好的指导原则来自动选择分量的个数,而是依赖于人为给出的模型假设,当这种假设并不符合实际情况时,就会使得这种参数式算法的表达能力受到严重的限制.然而值得注意的是:合理选择了高斯分量数目的GMM本身就具有很好的估计性能.基于这一观察,我们通过将GMM和在线学习这两者的优点结合起来,使得本文提出的算法能够根据输入数据的分布情况自适应地选择合适的分量数目,以及学习各个分量的参数.同时,模型的去燥调整过程赋予了它根据需要动态删除噪声高斯分量的能力,这不仅合理地控制了模型的复杂度,而且提高了模型在近似各种复杂分布时的灵活性.作为对上文所描述的密度估计学习过程的总结,算法 1 给出了完整的学习框架.

算法 1. 自适应在线核密度估计算法.

输入:数据流 $\{x | x \in R^d\}$, 参数$(\sigma,q,\eta,\lambda,v)$;

输出:混合模型 $\{C_i | C_i = (\mu_i,\Sigma_i,n_i,T_i)\}$.

1. 初始高斯分量 $C_0 \leftarrow (\mu_0 = x, \Sigma_0 = \sigma I, n_0 = 1, T_0 = \sqrt{\chi^2_{d,q}})$

2. 混合模型 $M \leftarrow \{C_1\}$
3. while $x \in R^d$ do
4. $D_i(x) = \sqrt{(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)}, i = 1,2,\ldots,K = |M|$
5. $S = \{i \mid i \in \{1,2,\ldots,K\}, D_i(x) < \eta T_i\}$
6. if $S=\varnothing$ then
7. $C_{new} \leftarrow (\mu_{new} = x, \Sigma_{new} = \sigma I, n_{new} = 1, T_{new} = \sqrt{\chi^2_{d,q}})$
8. $M \leftarrow M \cup \{C_{new}\}$
9. else
10. 根据公式(9)~公式(12)更新集合中的高斯分量
11. end if
12. if 当前学习样本数达到整数倍数 then
13. $S_{delete} = \{C_i | i \in \{1,2,\ldots,K\}, n_i < v\overline{n}\}$
14. 删除所有在 S_{delete} 中的高斯分量实现去燥
15. end while

3 算法分析

3.1 高斯分量的学习

基于竞争学习的神经网络能够实现在线学习,这样的神经网络,例如 SOM 模型[26]、ART 模型[27]、GNG 模型[28−31]、GCS 模型[32]以及 SOINN[1−10,14−42].它们都是基于竞争学习的在线学习算法,通过在输入空间中分布神经元达到学习数据分布的目的,每个神经元代表了其周围与它最相似的输入模式.从这一角度来看,这些算法本质上都是一种矢量量化[34]过程.每次新到来输入数据时,只有距离最近的神经元(通常称为获胜神经元)进行学习,每个胜者神经元的学习方式通常按如下方式进行权值更新:

$$w_i^t \leftarrow w_i^{t-1} + \epsilon(t)(x^t - w_i^{t-1}) \tag{15}$$

其中,参数 $\epsilon(t)$ 是学习率,t 为该神经元成为胜者的次数且 $\epsilon(t)$ 与 t 成反比,其意义在于使得神经元的移动最终能够趋于稳定,同时算法最终能够收敛.通常情况下,$\epsilon(t)$ 的形式需要满足一定的条件以保证学习的有效性.比如约束条件 $\sum_{t=1}^{\infty}\epsilon(t)=\infty, \sum_{t=1}^{\infty}\epsilon^2(t)<\infty$, 满足该约束条件的学习率可以保证每个神经元在逐渐稳定的情况下始终保持一定的学习能力[35].文献[36]对矢量量化的渐近性质作了分析并指出:在算法收敛后,量化向量 w 的密度 $\rho'(x)$ 与原数据集密度函数 $\rho(x)$ 具有如下关系 $\rho'(x) \propto \rho(x)^{\alpha}$, 其中,$\alpha$ 被称为放大因子(magnification factor),使用不同的算法来优化函数会得到不同的 α 值.

假设某神经元 i 从学习过程开始到 t 时刻共经历 n 次更新,根据以上的权值更新公式,当我们设置学习率 $\epsilon(t)=1/n$ 时,则有:

$$w_i^n \leftarrow \frac{1}{n}\sum_{t=1}^{n} x_t \tag{16}$$

因此,每个神经元最终的权值就是所有参与更新该神经元的训练样本的均值.从统计学的角度上来看,每个神经元保留了局部样本的均值信息.在此基础上,还可以为每个神经元增加局部协方差信息以保留更为丰富的数据分布信息.这样,每个神经元除了维护局部均值 μ 之外,还维护了一个局部协方差矩阵 Σ,该协方差矩阵保存了该神经元周围样本的分布信息.对于每个新的高斯分量,我们为其赋予初始值 $\Sigma=\sigma I$,这里,$\sigma>0$ 是一个正数.实际上,这可以看作是我们对要估计的协方差的每个方向上都增加了一个正则化系数.通过引入正则化系数,我们保证了估计得出的协方差矩阵一定是正定的:首先 Σ 本身是半正定的,然后对于 $\forall v \neq 0$, 我们都有:

$$v^T(\Sigma + \sigma I)v = v^T\Sigma v + \sigma v^T I v \geqslant 0 + \sigma v^2 > 0 \tag{17}$$

参数 σ 的作用实际上类似于核密度估计中的带宽参数,并且应当被设置得相对较小以避免过度平滑,导致

局部分布变化信息被忽略.但是,与核密度估计不同的是,即使σ设置得很小也不容易出现过拟合的情况,因为在我们的模型中,随着越来越多的新数据的输入,协方差矩阵会逐渐被更新到合适的值.因此,这样的处理方式能够增加算法在运行过程中的数值稳定性,随着训练数据的逐渐增多,可以让正则化系数σ逐渐趋近于0,使得对应的估计能够更加接近真实值.除此之外,我们还可以从贝叶斯推理的角度来解释这一做法的合理性,事实上,为协方差矩阵增加一个正则项对应了我们先验地认为协方差矩阵Σ服从高斯分布.

通过同时保留局部样本的均值和协方差信息,每个神经元实际上就维护了一个局部的高斯分布,而这种视角正是我们的在线密度估计算法的核心.此外,训练过程中每隔一段时间,距离相近而且分布相似的神经元将周期性地被合并以减少神经元的数量.这使得整个网络的复杂度始终得到有效的控制,增加了模型的稳定性.特别是,当数据近似地以高斯分布的形式存在时,这种方式几乎可以准确地恢复出所有聚类信息,当该条件不被满足时,由于该模型的自适应性以及多个高斯分布的近似能力,依然能够近似地还原出输入数据的分布.

3.2 高斯分量的扩张

在输入新数据 $x\in R^d$ 时,我们使用每个高斯分量的参数 T_i 来决定该高斯分量是否被激活,所以 T_i 的取值实际上决定了学习算法如何处理局部学习和全局学习的关系.如果假设每个局部的样本点都由一个高斯分布产生,那么根据马氏距离 $D_i(x)$的定义, $D_i^2(x)=(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)$ 就服从一个自由度为 d(样本特征维数)的 χ^2 分布 χ_d^2,并且其概率密度函数为

$$f(x)=\frac{1}{2^{d/2}\Gamma(d/2)}x^{d/2-1}\mathrm{e}^{-x/2} \tag{18}$$

设 χ_d^2 分布的上侧α分位点为 $\chi_{d,\alpha}^2$, 我们知道 $P\left[\chi^2\geqslant\chi_{d,\alpha}^2\right]=\alpha$. 因此,给定一个置信度 q,如果我们希望样本点落在马氏距离 $D_i(x)$所定义的超椭球体 $D_i^2(x)<T_i^2$ 内部的概率为 q,那么就可以表示为

$$P\left[D_i^2(x)<T_i^2\right]=P\left[(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)<T_i^2\right]=q \tag{19}$$

于是每个高斯分量的参数 T_i 可以取值为 $T_i=\sqrt{\chi_{d,q}^2}$, 其中,q 为 χ_d^2 分布的置信度(分位点),在实际应用中通常取较大值 0.95 或者 0.9,也可以将其从一个接近但小于 1 的值逐渐减小到 0.95 或者 0.9,这对于高斯分布或者 χ^2 分布都差不多是包含了距离均值两倍标准差范围内的分布空间.如果当前模型对数据拟合得不错,则基于高斯分布的假设,我们就有理由令 T_i为满足以上性质的值.然而,因为在线学习的缘故,每一个高斯分量都需要足够多的样本才能收敛到理想的状态.因此,我们在实际应用时选择 $T_i=\eta_{n_i}\sqrt{\chi_{d,q}^2}$, 这里, $\eta_{n_i}\geqslant 1$是一个与高斯分量的参数 n_i有关的因子,并且随着 n_i的增大而减小.这个策略可以使得每个高斯分量在初始阶段能够有扩张的趋势,以包含更多的数据点进行自身参数的学习.随着学习数据的增多, η_{n_i} 会逐渐减小并趋向于 1,T_i 也会逐渐趋向于 $\chi_{d,q}^2$, 这样,每个高斯分量可以达到一种稳定的学习状态.在本文的实验部分,我们取 $\eta_{n_i}=1+1.05^{1-n_i}$.

3.3 与高斯混合模型的关系

用一定数量的代表性数据来近似原始的完整数据集,然后在这些代表点的基础上对新来的数据做出决策是基于原型(prototype-based)的学习算法的基本思想.这些代表性的数据被称为原型(prototype),它们通常以某种方式反映了原始数据的分布信息,比如 *k*-means[33]的各个聚类中心就是一种典型的原型.*k*-means 算法用于聚类时只使用了样本的均值信息,然而我们也可以为各个聚类中心计算其周围样本的协方差,然后认为每个聚类都是一个高斯分布,所有的聚类就代表了整体分布的各个高斯分量.在此基础上,我们也可以用这些高斯分量进行整体的密度估计,于是就从 *k*-means 得到了一个 GMM.事实也确实如此,从理论上可以证明:*k*-means 算法的运行过程实际上可以看作是 EM 算法应用于 GMM 时的特殊情况.然而,*k*-means 和 GMM 都是离线式的学习算法,并且需要保留所有的原始训练数据,这样应用到大规模数据集上的计算代价非常高.

从某种程度上可以说,SOINN 是对 *k*-means 算法的一种增量式实现方式,我们提出的算法则可以认为是在 SOINN 的基础上增加了数据分布的协方差信息,使得每个 SOINN 神经元对数据分布的表达能力更为丰富.从模型的最终结果来看,我们的算法实际上就成为了一种增量式的 GMM.每个神经元都是一个动态的高斯分量,拟

合了它周围的训练样本,而且可以动态生成也可以被动态删除,具有了随着数据分布而适应性变化的灵活性.而训练该模型的方法就是对每个神经元的参数分别做最大似然估计,由于每输入一个样本,模型参数都能够得到更新,因此相比于使用 EM 算法训练的 GMM,我们算法的训练速度更快,收敛需要的样本数量也大为减少.当新的样本数据x进来时,传统的EM算法需要对当前模型的每个高斯分布都作一个全局的更新,这个操作可以保证增加数据的期望似然值.然而,这种针对全局的更新操作很容易破坏先前已经学习到的模型结构,而且容易陷入到局部最优值,尤其是当新输入数据 x 不服从之前已经学习到的分布时.同时,如果 x 只是一个噪音点,那么任何针对该噪音点的参数更新都会导致不好的结果.这与同样以竞争学习为基础的 SOMN 有很大不同,SOMN 以 SOM 为基础框架提出了一个类似的混合模型,但是,由于 SOM 模型自身的特点,作为混合分量载体的神经元之间的拓扑结构需要预先设定,而且形式上也受到相当大的限制,因此并不能很好地反映原始数据的分布情况.我们的算法则没有这个限制,通过为每一个高斯分量设定一个自适应的相似度阈值参数,可以动态地增加或者删除分量来适应新的数据,同时不破坏先前已经学习到的分布信息.

4 实验结果

为了验证算法的有效性,我们在人工数据集和真实数据集上进行了密度估计实验,并与现有的密度估计算法进行了比较.这些算法包括:核密度估计(kernel density estimation),其带宽参数为高斯分布情况下的最优值;使用基于数据的启发式方法进行带宽参数选择的核密度估计(kernel density estimation with bandwidth selection);高斯混合模型(Gaussian mixture model);鲁棒的核密度估计(robust kernel density estimation);规约集密度估计器(reduced set density estimator);基于扩散过程的核密度估计(kernel density estimation via diffusion);在线核密度估计(online kernel density estimation).其中,只有 oKDE 和我们提出的算法能够实现在线学习,其他算法都只能离线训练.表 2 给出了这些算法的概要信息.

Table 2 Comparison algorithm for density estimation

表 2 密度估计对比算法

序号	名称	密度估计方法	类型
1	KDE	Kernel density estimation	Offline
2	KDEb	Kernel density estimation with bandwidth selection	Offline
3	GMM	Gaussian mixture model	Offline
4	RKDE	Robust kernel density estimation	Offline
5	RSDE	Reduced set density estimator	Offline
6	KDEd	Kernel density estimation via diffusion	Offline
7	oKDE	Online kernel density estimation	Online
8	(Proposed)	Adaptive online kernel density estimation	Online

4.1 人工数据

我们首先在人工数据集上进行了密度估计对比实验.为便于观察估计结果的准确性,我们测试了以下数据集:2 个一维高斯分布组成的双模(bimodal)分布;6 个一维高斯分布组成的爪形(claw)分布以及 3 个二维高斯分布组成的混合(mixture)分布.这些数据集的分布函数和具体信息见表 3.

Table 3 Information about artificial data set

表 3 人工数据集信息

序号	形状	概率密度函数	维数
1	Bimodal	$\frac{1}{2}N\left(0,\left(\frac{1}{10}\right)^2\right)+\frac{1}{2}N(5,1)$	1
2	Claw	$\frac{1}{2}N(0,1)+\sum_{k=0}^{4}\frac{1}{10}N\left(\frac{k}{2}-1,\left(\frac{1}{10}\right)^2\right)$	1
3	Mixture	$\frac{1}{2}N\left(\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)+\frac{3}{10}N\left(\begin{bmatrix}3\\3\end{bmatrix},\begin{bmatrix}2&0\\0&0.5\end{bmatrix}\right)+\frac{2}{10}N\left(\begin{bmatrix}5\\0\end{bmatrix},\begin{bmatrix}0.5&0\\0&2\end{bmatrix}\right)$	2

我们提出的在线密度估计算法中涉及到的参数组合为(σ,q,η,λ,v),在实验中设置q=0.8,λ=1000,v=0.1,这3个参数的取值具有通用性,一般不依赖于具体的问题场景.正如前面所分析的,参数的取值会影响高斯分量的初始更新行为,在所有的实验中设置为$\eta_{n_i}=1+1.05^{1-n_i}$,而参数则比较依赖于具体的数据特性,在 3 个人工数据实验中分别取值 0.3、0.1、0.5.

首先是密度估计算法评估中常用的双模分布(表 3 中第 1 个数据集),训练集由 3 000 个按其概率密度函数抽取的样本组成,各种估计算法得到的密度曲线如图 2 所示(图中的黑色曲线对应真实概率密度函数,而红色曲线则对应于各种密度估计算法得到的概率密度函数).

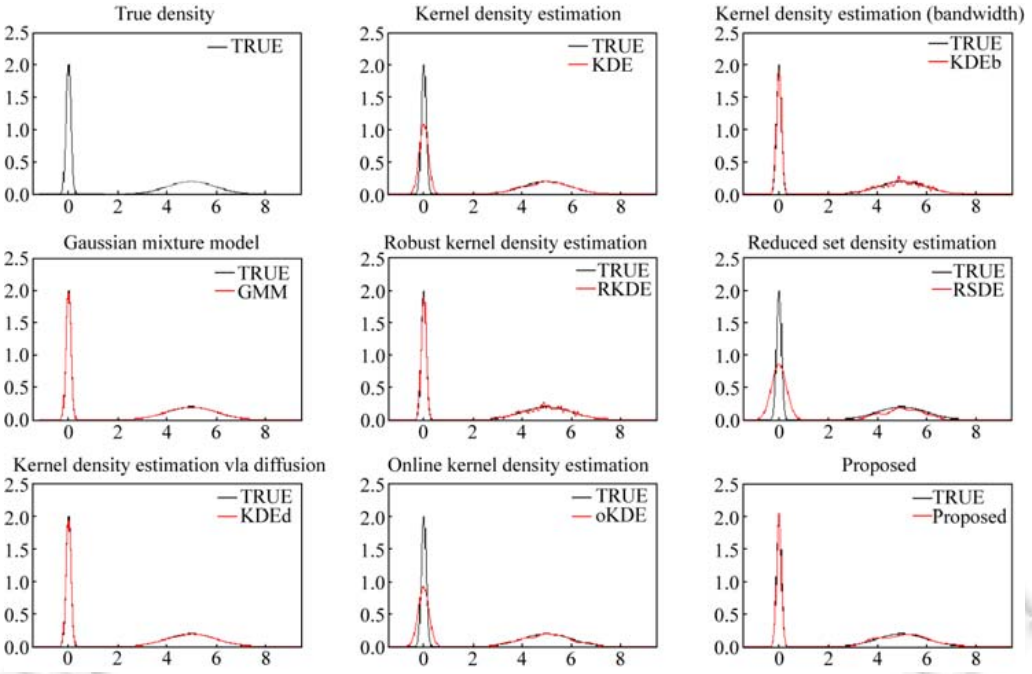

Fig.2 Experimental results of bimodal distribution

图 2 双模分布实验结果

从密度曲线结果图中我们可以初步看到,GMM、KDEd 以及我们提出的算法准确地恢复出了真实的密度曲线.而 KDEb 和 RKDE 虽然对左侧分布比较集中的高斯分布拟合得很好,但是对于右边方差较大的高斯分布的估计曲线中锯齿状非常明显,这是因为,这两种方法都是直接指定了一个全局的带宽参数,所以并不能很好地适应分布的局部变化.而 KDE 和 RSDE 以及 oKDE 都较好地拟合了右侧方差较大的高斯分布,但是它们都不能有效地对左侧方差较小的高斯分布进行估计.因此,与同类密度估计算法相比,我们的算法得到了与最好的离线密度估计算法相当的实验效果.

再看另一个局部分布结构更加复杂多变的爪形分布(表 3 中的第 2 个数据集),训练集同样是由给定的密度函数抽取的 3 000 个样本组成,实验中 GMM 模型的超参数(即高斯分量个数)设置为 6(即真实高斯分量个数),各种算法得到的密度曲线如图 3 所示(图中的黑色曲线对应真实概率密度函数,而红色曲线则对应于各种密度估计算法得到的概率密度函数).从该密度曲线图中我们可以初步看到,KDEb 和 KDEd 在复杂分布情况下很好地刻画了真实的密度曲线.虽然我们使用了最优的超参数,但是 GMM 并没有捕捉到局部密度变化,出现欠拟合的现象.RKDE 和 RSDE 虽然对中间复杂变化的分布估计较好,但是对于两侧的分布,RKDE 几乎没有进行拟合,RSDE 的拟合曲线的锯齿状则比较明显.oKDE 的估计结果非常平滑,未能很好地捕捉到局部的密度变化.最后,我们提出的算法对中间变化较复杂的曲线部分拟合效果基本符合真实密度分布,对两侧的分布也较好地进

行了处理.因此与其他算法相比,我们的算法同样实现了很好的密度估计结果.

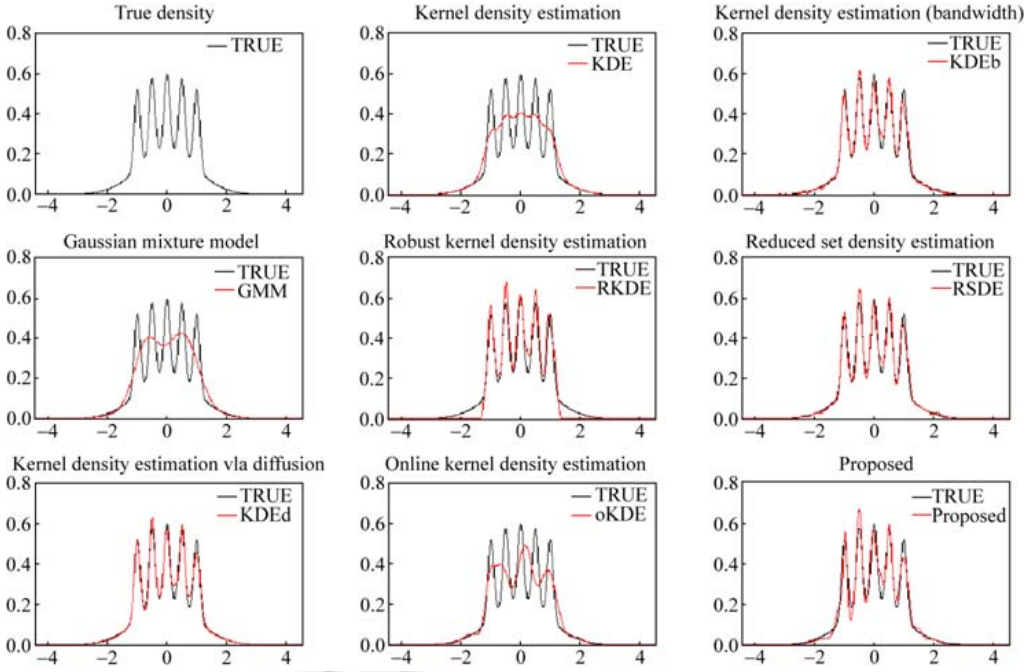

Fig.3 Experimental results of claw distribution

图 3 爪形分布实验结果

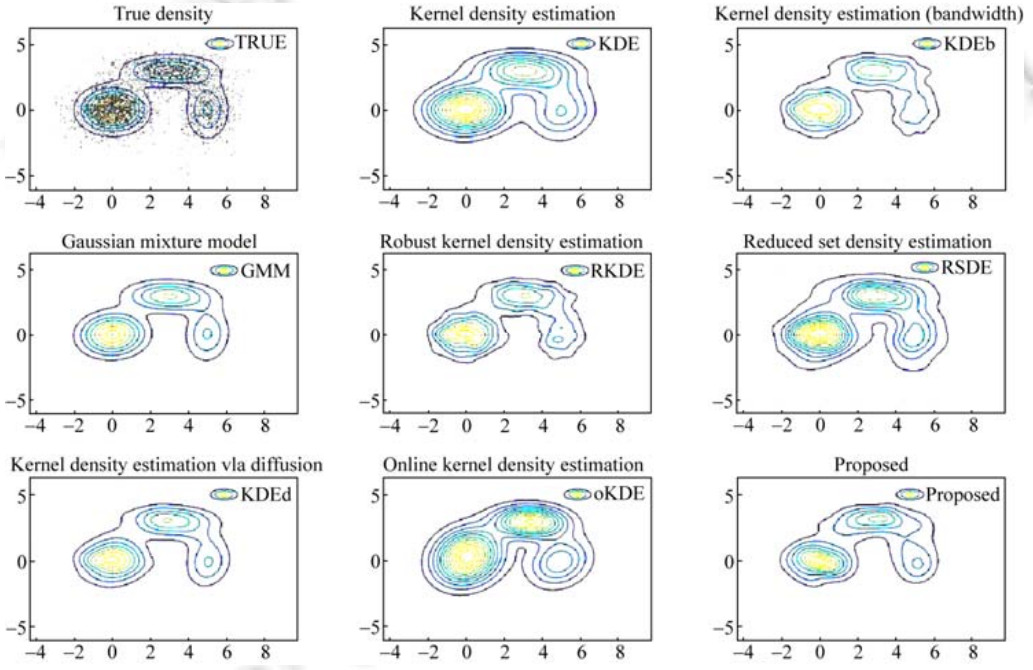

Fig.4 Experimental results of mixture distribution

图 4 混合分布实验结果

除了以上两个典型的用于密度估计的一维概率密度函数以外,我们还在二维数据集上进行了密度估计对

比实验,以进一步验证密度估计算法的性能.该数据集是由 3 个高斯分布组成的高斯混合分布(表中的第 3 个数据集).从实验结果可以看到,因为使用了最优参数,所以 GMM 的估计结果确实非常接近真实分布.KDEb 和 KDEd 也得到了很好的估计性能.值得注意的是,oKDE 因为其基于全局卷积操作的特性,与一维情况类似,出现了过度平滑的现象.我们提出的在线密度估计算法则已经非常接近最优的离线算法 KDEd,并且比现有的在线核密度估计算法 oKDE 的估计结果更加准确.

Table 4 NLL value of artificial data set

表 4 人工数据集上的负对数似然值(NLL)

Data set	KDE	KDEb	GMM	RKDE	RSDE	KDEd	oKDE	Proposed
Bimodal	0.986 4	0.950 3	**0.925 5**	0.950 4	1.440 2	**0.925 6**	1.117 0	**0.934 7**
Claw	1.262 0	1.212 6	1.281 5	2.307 0	**1.198 1**	**1.195 1**	1.285 8	**1.207 5**
Mixture	3.826 5	**3.778 8**	**3.763 5**	3.854 6	3.913 7	**3.768 9**	3.813 7	3.787 8

针对这些密度估计算法具体的性能比较,我们首先使用各种算法在训练集上得到估计模型,然后在测试集上使用平均负对数似然值(negative log-likelihood,简称 NLL)作为估计算法的评价准则.该值越低,表明模型对于测试数据集似然函数越大,亦即对于密度估计的平均置信度越高,从这个角度来说,估计效果就越好.各种密度估计算法在以上 3 个人工数据集上的实验结果见表 4(粗体标明了最优的前 3 个值,其中添加下划线的为最优值).从中可以看出,具体的数值比较结果基本上符合上文对各种算法的定性分析.其中,GMM 模型在双峰分布和混合分布中取得最好的结果,这是符合预期的,因为我们使用了最优的高斯分量数目假设,也侧面反映了 NLL 作为密度估计评价指标的有效性.KDEd 则在 3 个测试分布上都取得了非常好的估计结果,体现了其离线情况下优异的密度估计性能.而我们提出的算法也在两个分布上取得了与最好算法相当的实验结果,特别是要比同为在线算法的 oKDE 的估计结果更好.综合以上各算法在人工数据集上的实验结果,初步验证了我们提出的自适应在线密度估计算法的有效性.

4.2 真实数据

我们还在真实数据集上进行了密度估计对比实验,使用的数据集来自于 UCI 机器学习库[43]和 LIBSVM[44],具体信息见表 5 和表 6.其中,表 5 包含了一些代表性数据集,虽然它们的样本数量不是很多,但是便于我们评估各种算法的有效性并进行多种算法的对比.同时,为了突出在线密度估计算法的特点,我们使用表 6 中包含的数据集来验证 oKDE 和本文提出的算法在大规模数据集上的密度估计性能.

针对表 5 中包含的数据集,参与对比实验的密度估计算法和在人工数据集上的相同,并且包括密度估计和基于密度估计的分类两种实验设置.对于每个数据集,我们随机选择 70%的样本用作训练集,剩下的 30%用作测试集.对于密度估计实验任务,我们以 NLL 作为评价准则,得到的结果见表 7(粗体标明了最优的前 3 个值,其中添加下划线的为最优值).从表中可以看到,我们的算法基本上实现了与 KDEb 以及 GMM(使用最优超参数)差不多的估计性能,并且在大部分数据集上都要比同样是在线算法的 oKDE 估计的结果更好.对于分类任务,我们使用分类准确率作为评价准则,得到的结果见表 8(粗体标明了最优的前 3 个值,其中添加下划线的为最优值).与判别式模型不同,基于密度估计的分类算法可以作为生成式分类模型,即我们首先估计各个类别条件概率,然后结合先验分布,按照贝叶斯公式得到类别的后验概率.从实验结果中可以看出,我们的算法在大部分真实数据集上得到了最优或者次优的分类准确率,并且好于在线密度估计算法 oKDE.

Table 5 Information about real data set

表 5 真实数据集信息

Data set	# of class	# of instance	# of feature
Iris	3	150	4
Wine	3	178	13
Glass	6	214	9
Diabetes	2	768	8
Breast cancer	2	683	10
Image segmentation	7	2310	19

Table 6 Information about real large scale data set

表 6 真实大规模数据集信息

Data set	# of class	# of instance	# of feature
Letter	26	20 000	16
Shuttle	7	58 000	9
Webb Spam	2	350 000	254

Table 7 NLL value of real data set

表 7 真实数据集上的负对数似然值(NLL)

Data set	KDE	KDEb	GMM	RKDE	RSDE	KDEd	oKDE	Proposed
Iris	4.402 4	**–0.049 6**	1.303 9	**–0.049 8**	0.400 5	0.181 9	0.570 0	**0.073 6**
Wine	13.555 7	**2.639 6**	4.100 2	**2.639 6**	4.628 5	5.307 3	4.460 7	**3.899 8**
Glass	9.019 7	0.776 9	1.081 9	0.776 9	0.524 2	**0.400 1**	**–5.195 2**	**–3.076 6**
Diabetes	8.087 6	**1.084 3**	2.560 0	**1.084 3**	2.213 3	1.389 8	2.129 3	**1.194 8**
Cancer	10.828 3	**0.023 9**	3.425 0	**0.023 9**	2.866 3	2.774 2	1.603 6	**–1.636 1**
Segment	18.010 2	**–5.022 4**	2.153 3	–5.022 4	–0.363 2	–0.480 6	**–24.624 6**	**–27.135 7**

Table 8 Accuracy of real data set

表 8 真实数据集上的分类准确率

Data set	KDE	KDEb	GMM	RKDE	RSDE	KDEd	oKDE	Proposed
Iris	0.937 8	0.955 6	**0.982 2**	0.951 1	0.973 3	**0.977 8**	0.968 9	**0.977 8**
Wine	**0.978 2**	**0.981 7**	0.974 5	**0.981 8**	0.956 4	0.970 9	0.916 4	0.974 5
Glass	0.615 1	0.698 1	**0.766 0**	0.698 1	**0.739 6**	0.637 7	**0.739 4**	0.701 9
Diabetes	0.649 4	0.716 0	**0.726 4**	0.716 0	**0.731 6**	0.719 5	0.699 6	**0.748 9**
Cancer	0.939 8	0.672 8	**0.967 0**	0.672 8	0.944 7	**0.972 8**	0.925 2	**0.957 3**
Segment	0.839 4	0.569 4	0.885 1	0.569 4	**0.919 1**	0.842 9	**0.909 1**	**0.908 3**

Table 9 Training time of real data set (s)

表 9 真实数据集上的算法训练时间 (秒)

Data set	KDE	KDEb	GMM	RKDE	RSDE	KDEd	oKDE	Proposed
Iris	N/A	N/A	0.015 7	0.002 7	0.002 0	0.046 7	0.915 4	0.034 1
Wine	N/A	N/A	0.011 9	0.001 2	0.002 4	0.110 5	2.171 1	0.041 2
Glass	N/A	N/A	0.045 2	0.005 9	0.002 0	0.040 0	1.948 1	0.032 5
Diabetes	N/A	N/A	0.006 6	0.020 9	0.003 4	0.282 1	37.306 5	0.209 5
Cancer	N/A	N/A	0.004 9	0.008 3	0.001 7	0.163 3	31.380 3	0.239 0
Segment	N/A	N/A	0.124 5	0.267 3	0.022 1	5.806 8	305.403 3	0.653 1

为了比较各种算法的实践应用能力,我们还给出了它们的运行时间对比,结果见表 9.其中,KDE 和 KDEb 因为实际上并没有学习过程,所以其训练时间表示为 N/A(not available),或者也可以认为它们的训练时间为 0.但是它们所要求的存储空间和测试时间比其他算法要多,这实际上也是 KDE 类算法在实际应用中最需要解决的问题之一.从表中所示结果可以看到,GMM、RKDE、RSDE 的训练时间基本上差不多,KDEd 因为额外地自适应选择带宽参数的过程,需要的训练时间要比它们长,在线算法 oKDE 则计算非常耗时,尤其是在特征维数比较高的时候,而本文提出的在线密度估计算法却比 oKDE 快得多.对于在线算法,受限于模型的动态变化,不太容易给出准确的运行时间,但是如果我们假设高斯分量数目在收敛之前呈现线性增长,且最终的高斯分量数目为 K,同时假设学习的样本数为 N,样本的特征维数为 d,那么我们算法的时间复杂度可以估计为 $O(NK^2d^3)$,其中,因子 d^3 是每个样本的更新复杂度(假设每次激活的高斯分量数为常量),因子 NK 则是估计的更新次数.

为了验证在线密度估计算法在大规模数据集上的估计性能,我们还对表 6 中包含的数据集进行了分类实验.考虑到计算复杂度的因素和出于同类算法对比的目的,在这部分实验中只比较了在线算法 oKDE 和本文提出的算法,得到的结果见表 10,包含了分类准确率和训练所需时间,其中,N/A 表示对应的算法在较长一段时间(5 天)内还没有给出学习结果.对于 Letter 数据集,我们的算法要比 oKDE 分类准确率好一点,但是训练时间却要少得多;对于 Shuttle 数据集,我们算法的分类准确率和训练时间都要比 oKDE 好很多;对于 Webb Spam 数据集,oKDE 的训练时间非常长,至少 5 天时间内仍然没有给出结果,而我们的算法很快就可以得到一个不错的分类准确率.所以相对而言,本文提出的算法更适合应用于大规模数据集,具有更好的实践意义.

Table 10 Accuracy/Training time of real large scale data set (s)
表 10 真实大规模数据集上的分类准确率/训练时间 (秒)

Data set	oKDE	Proposed
Letter	0.4517/1420.4	0.4562/1.2119
Shuttle	0.7978/107.14	0.9460/0.4662
Webb Spam	N/A	0.8436/~1h

5 结束语

本文提出了一种新的在线密度估计算法,该算法结合了高斯混合模型和在线学习的优点,通过最大化每个局部高斯分量周围训练数据的似然值,对模型参数进行更新,具有在线学习数据概率分布密度的能力.同时,它又可以被认为是一种增量的 GMM.通过为每个高斯分量设定一个具有统计意义的相似度阈值参数,使得每个高斯分量只对落在自身周围的数据样本进行学习.结合高斯分量的相似度阈值参数,能够在不破坏先前已经学到的模型分量的基础上,动态地增加或者删除高斯分量来适应新的数据分布.这种局部学习策略给密度估计过程带来了很多好处:首先,与当前输入数据相关性较小的高斯分量不会在参数更新中受到影响,这也就意味着这些分量在之前所学习到的分布信息能够得到保留,而不会因为新数据的分布变化被破坏;其次,由于每次只有一部分高斯分量需要进行更新,因此减少了相应的计算代价;最后,因为所采用的更新步长以每个分量对当前学习样本的贡献度为权重进行调整,因此这种局部学习会比全局学习具有相对更快的更新,这也在一定程度上加速了整个模型的收敛过程.

因为在线学习的特点,我们提出的算法尤其适合于大规模数据集上的密度估计和流式计算任务[45],而且能够处理分布不断变化的非稳态数据.在人工数据集和真实数据集上的实验结果表明:我们提出的算法能够取得与现有最好的离线学习算法差不多的密度估计性能,同时优于现有的在线密度估计算法.并且在学习过程中,我们的模型不需要存储任何历史训练数据,而是仅仅保留了当前学习到的最优模型,因此,它的时间复杂度和空间复杂度比传统的离线学习算法要低得多.

References:

[1] Latecki LJ, Lazarevic A, Pokrajac D. Outlier detection with kernel density functions. In: Proc. of the Int'l Workshop on Machine Learning and Data Mining in Pattern Recognition. Berlin, Heidelberg: Springer-Verlag, 2007. 61–75.

[2] Laxhammar R, Falkman G, Sviestins E. Anomaly detection in sea traffic-a comparison of the gaussian mixture model and the kernel density estimator. In: Proc. of the 12th Int'l Conf. on Information Fusion. 2009. 756–763.

[3] Costa BSJ, Angelov PP, Guedes LA. Real-time fault detection using recursive density estimation. Journal of Control, Automation and Electrical Systems, 2014,25(4):428–437. https://doi.org/10.1007/s40313-014-0128-4

[4] Chen H, Meer P. Robust computer vision through kernel density estimation. In: Proc. of the 2002 European Conf. on Computer Vision. Berlin, Heidelberg: Springer-Verlag, 2002. 236–250. https://doi.org/10.1007/3-540-47969-4_16

[5] Yang C, Duraiswami R, Gumerov NA, Davis L. Improved fast Gauss transform and efficient kernel density estimation. In: Proc. of the IEEE Int'l Conf. on Computer Vision. 2003. 664–671. https://doi.org/10.1109/ICCV.2003.1238383

[6] Elgammal A, Duraiswami R, Harwood D, *et al*. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. In: Proc. of the IEEE. 2002. 1151–1163. https://doi.org/10.1109/JPROC.2002.801448

[7] Mittal A, Paragios N. Motion-based background subtraction using adaptive kernel density estimation. In: Proc. of the 2004 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition. 2004. 302–309. https://doi.org/10.1109/CVPR.2004.1315179

[8] Zivkovic Z, Ferdinand VDH. Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognition Letters, 2006,27(7):773–780. https://doi.org/10.1016/j.patrec.2005.11.005

[9] Nakaya T, Yano K. Visualising crime clusters in a space-time cube: An exploratory data-analysis approach using space-time kernel density estimation and scan statistics. Trans. in GIS, 2010,14(3):223–239. https://doi.org/10.1111/j.1467-9671.2010.01194.x

[10] Lampe OD, Hauser H. Interactive visualization of streaming data with kernel density estimation. In: Proc. of the 2011 IEEE Pacific Visualization Symp. 2011. 171–178. https://doi.org/10.1109/PACIFICVIS.2011.5742387

[11] McLachlan G, Peel D. Finite Mixture Models. New York: John Wiley & Sons, 2000. https://doi.org/10.1002/0471721182

[12] Qiu TY. Research of online density estimation based on incremental Gaussian mixtures [MS. Thesis]. Nanjing: Nanjing University, 2016 (in Chinese with English abstract).

[13] Parzen E. On estimation of a probability density function and mode. The Annals of Mathematical Statistics, 1962,33(3):1065–1076.

[14] Vapnik V, Mukherjee S. Support vector method for multivariate density estimation. In: Advances in Neural Information Processing Systems. 2000. 659–665.

[15] Girolami M, He C. Probability density estimation from optimally condensed data samples. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2003,25(10):1253–1264.

[16] Kim JS, Scott CD. Robust kernel density estimation. Journal of Machine Learning Research, 2012,13(Sep):2529–2565.

[17] Botev ZI, Grotowski JF, Kroese DP. Kernel density estimation via diffusion. The Annals of Statistics, 2010,38(5):2916–2957.

[18] Yin H, Allinson NM. Self-organizing mixture networks for probability density estimation. IEEE Trans. on Neural Networks, 2001, 12(2):405–411.

[19] Vlassis N, Likas A. A greedy EM algorithm for Gaussian mixture learning. Neural Processing Letters, 2002,15(1):77–87. https://doi.org/10.1023/A:1013844811137

[20] Han B, Comaniciu D, Davis L. Sequential kernel density approximation through mode propagation: Applications to background modeling. In: Proc. of the ACCV. 2004. 818–823.

[21] Cheng Y. Mean shift, mode seeking, and clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1995,17(8): 790–799.

[22] Song M, Wang H. Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering. In: Proc. of the Intelligent Computing: Theory and Applications III. Int'l Society for Optics and Photonics, 2005. 174–184. https://doi.org/10.1117/12.601724

[23] Engel PM, Heinen MR. Incremental learning of multivariate gaussian mixture models. In: Proc. of the Brazilian Symp. on Artificial Intelligence. Berlin, Heidelberg: Springer-Verlag, 2010. 82–91. https://doi.org/10.1007/978-3-642-16138-4_9

[24] Kristan M, Leonardis A, Skočaj D. Multivariate online kernel density estimation with Gaussian kernels. Pattern Recognition, 2011,44(10–11):2630–2642. https://doi.org/10.1016/j.patcog.2011.03.019

[25] Kristan M, Skočaj D, Leonardis A. Online kernel density estimation for interactive learning. Image and Vision Computing, 2010,28(7):1106–1116. https://doi.org/10.1016/j.imavis.2009.09.010

[26] Kohonen T. Self-organized formation of topologically correct feature maps. Biological Cybernetics, 1982,43(1):59–69. https://doi.org/10.1007/BF00337288

[27] Carpenter GA, Grossberg S. The ART of adaptive pattern recognition by a self-organizing neural network. Computer, 1988,21(3): 77–88. https://doi.org/10.1109/2.33

[28] Martinetz TM. A "neural-gas" network learns topologies. Artificial Neural Networks, 1991,1(1):397–402.

[29] Martinetz TM, Berkovich SG, Schulten KJ. "Neural-gas" network for vector quantization and its application to time-series prediction. IEEE Trans. on Neural Networks, 1993,4(4):558–569. https://doi.org/10.1109/72.238311

[30] Martinetz T, Schulten K. Topology representing networks. Neural Networks, 1994,7(3):507–522.

[31] Fritzke B. A growing neural gas network learns topologies. In: Advances in Neural Information Processing Systems. 1995. 625–632.

[32] Fritzke B. Growing cell structures—a self-organizing network for unsupervised and supervised learning. Neural Networks, 1994, 7(9):1441–1460.

[33] MacQueen J. Some methods for classification and analysis of multivariate observations. In: Proc. of the 5th Berkeley Symp. on Mathematical Statistics and Probability. 1967,1(14):281–297.

[34] Gray R. Vector quantization. IEEE ASSP Magazine, 1984,1(2):4–29.

[35] Robbins H, Monro S. A stochastic approximation method. In: Herbert Robbins Selected Papers. New York: Springer-Verlag, 1985. 102–109.

[36] Zador P. Asymptotic quantization error of continuous signals and the quantization dimension. IEEE Trans. on Information Theory, 1982,28(2):139–149.

[37] Terrell GR, Scott DW. Variable kernel density estimation. The Annals of Statistics, 1992,20(3):1236–1265.

[38] Burges CJC. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 1998,2(2): 121–167. https://doi.org/10.1023/A:1009715923555

[39] Shen F, Hasegawa O. An incremental network for on-line unsupervised classification and topology learning. Neural Networks, 2006,19(19):90–106.

[40] Shen F, Ogura T, Hasegawa O. An enhanced self-organizing incremental neural network for online unsupervised learning. Neural Networks, 2007,20(8):893–903.

[41] Shen F, Hasegawa O. Self-organizing incremental neural network and its application. In: Proc. of the Int'l Conf. on Artificial Neural Networks. Berlin, Heidelberg: Springer-Verlag, 2010. 535–540. https://doi.org/10.1007/978-3-642-15825-4_74

[42] Qiu TY, Shen FR, Zhao JX. Review of self-organizing incremental neural network. Ruan Jian Xue Bao/Journal of Software,2016,27(9):2230−2247 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5068.htm [doi: 10.13328/j.cnki.jos.005068]

[43] Bache K, Lichman M. UCI machine learning repository, 2013. http://archive.ics.uci.edu/ml

[44] Chang CC, Lin CJ. LIBSVM: A library for support vector machines. ACM Trans. on Intelligent Systems and Technology, 2011,2(3):27:1–27:27. https://doi.org/10.1145/1961189.1961199

[45] Sun DW, Zhang GY, Zheng WM. Big data stream computing: Technologies and instances. Ruan Jian Xue Bao/Journal of Software, 2014,25(4):839−862 (in Chinese). http://www.jos.org.cn/1000-9825/4558.htm [doi: 10.13328/j.cnki.jos.004558]

附中文参考文献:

[12] 邱天宇.基于增量高斯混合模型的在线密度估计研究[硕士学位论文].南京:南京大学,2016.

[42] 邱天宇,申富饶,赵金熙.自组织增量学习神经网络综述.软件学报,2016,27(9):2230−2247. http://www.jos.org.cn/1000-9825/5068.htm [doi: 10.13328/j.cnki.jos.005068]

[45] 孙大为,张广艳,郑纬民.大数据流式计算:关键技术及系统实例.软件学报,2014,25(4):839−862. http://www.jos.org.cn/1000-9825/4558.htm [doi: 10.13328/j.cnki.jos.004558]

邓齐林(1990－),男,安徽芜湖人,硕士,主要研究领域为神经网络,机器学习.

申富饶(1973－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为神经计算,机器人智能.

邱天宇(1991－),男,硕士,主要研究领域为机器学习,数据挖掘.

赵金熙(1950－),男,博士,教授,博士生导师,主要研究领域为计算数学,大规模科学计算,计算智能.