计算机科学
COMPUTER SCIENCE

三维流场的流线深度特征学习与特征聚类

陈杰, 金林江, 郑红波, 秦绪佳

引用本文

陈杰, 金林江, 郑红波, 秦绪佳. 三维流场的流线深度特征学习与特征聚类[J]. 计算机科学, 2024, 51(7): 221-228.

CHEN Jie, JIN Linjiang, ZHENG Hongbo, QIN Xujia. Deep Feature Learning and Feature Clustering of Streamlines in 3D Flow Fields [J]. Computer Science, 2024, 51(7): 221-228.

相似文章推荐（请使用火狐或 IE 浏览器查看文章）

Similar articles recommended (Please use Firefox or IE to view the article)

一种基于属性相似性和分布结构连通性的聚类算法

Clustering Algorithm Based on Attribute Similarity and Distributed Structure Connectivity

计算机科学, 2024, 51(7): 124-132. https://doi.org/10.11896/jsjkx.231000125

基于机器学习的异常流量检测模型优化研究

Study on Optimization of Abnormal Traffic Detection Model Based on Machine Learning

计算机科学, 2024, 51(6A): 230700051-5. https://doi.org/10.11896/jsjkx.230700051

基于残差密集卷积自编码的高噪声图像去噪方法

Residual Dense Convolutional Autoencoder for High Noise Image Denoising

计算机科学, 2024, 51(6A): 230400073-7. https://doi.org/10.11896/jsjkx.230400073

基于聚类优化学习的少样本图像分类

Few-shot Images Classification Based on Clustering Optimization Learning

计算机科学, 2024, 51(6A): 230300227-7. https://doi.org/10.11896/jsjkx.230300227

基于集成学习的跨语言文本主题发现方法研究

Cross-lingual Text Topic Discovery Based on Ensemble Learning

计算机科学, 2024, 51(6A): 230300201-8. https://doi.org/10.11896/jsjkx.230300201

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230500033

三维流场的流线深度特征学习与特征聚类

陈　杰　　金林江　　郑红波　　秦绪佳
浙江工业大学计算机科学与技术学院　杭州 310023
(987458138@qq.com)

摘　要　流场可视化指将流体运动的数据转换为视觉形式,以便更好地理解和分析流场的流动。利用流线来实现流场可视化,是当前最为热门的方法。文中提出了一种学习、聚类三维流场流线特性的方法。首先设计了一种基于卷积的自编码器来提取流线特征。该方法中的自编码器由编码器和解码器组成,其中编码器用卷积层降维的方式来提取输入流线的特征,而解码器使用转置卷积对流线特征进行上采样,以此重建流线。通过训练不断减小输入流线与重建流线的差异,可以让编码器提取到的流线特征更加准确。其次,改进了CFSFDP算法,用于流线特征聚类。针对原CFSFDP算法需要手动选取聚类中心,以及对距离参数过于敏感的缺点,改进了其指标计算方法,实现对聚类中心的自动选取,并且引入了高斯核密度估计,实现对截断距离参数的自适应计算。实验结果表明,所提方法在流线特征的学习以及聚类上具有良好的效果。

关键词:流场可视化;流线特征;卷积自编码器;聚类
中图分类号　TP391

Deep Feature Learning and Feature Clustering of Streamlines in 3D Flow Fields

CHEN Jie,JIN Linjiang,ZHENG Hongbo and QIN Xujia
College of Computer Science and Technology,Zhejiang University of Technology,Hangzhou 310023,China

Abstract　Flow field visualization refers to converting data of fluid motion into visual forms for better understanding and analysis of flow in the field. Using streamlines for flow field visualization is currently the most popular method. This paper proposes a method for learning and clustering 3D flow field streamline features. Firstly, a convolutional autoencoder-based method is designed to extract streamline features. The autoencoder in this method consists of an encoder and a decoder. The encoder uses convolutional layers to reduce the dimensions of input streamlines to extract features, while the decoder uses transpose convolution to upsample the streamline features to restore the streamlines. By continuously reducing the difference between input and restored streamlines through training, the encoder can extract more accurate streamline features. Secondly, this paper improves the CFSFDP(clustering by fast search and find of density peaks) algorithm for clustering streamline features. To address the issue of manually selecting cluster centers and the problem of sensitivity to distance parameters in the CFSFDP algorithm, this paper improves its metric calculation method, realizes automatic selection of cluster centers, and introduces adaptive calculation of truncation distance parameters using Gaussian kernel density estimation. Experimental results show that this method has good performance in learning streamline features and clustering.

Keywords　Flow field visualization, Streamline feature, Convolutional autoencoder, Clustering

1　引言

流场是一种描述流体(液体或气体)在空间中运动的物理现象的概念。它包含了在某一时间内,流体在各个点上的速度、压力、密度等物理量,流场可以用矢量场的形式进行表示。流场通常用于分析流体的行为和运动,以及对流体进行优化和控制的设计和实现。流场可视化方法是一种用于展示流场的可视化技术,根据其视觉表示可分为不同类型,包括直接可视化、基于纹理的可视化、基于集合的可视化和基于特征的可视化等。同时,流场可视化还可以根据数据的维度分为二维和三维流场可视化两大类。本文的研究工作主要集中在采用三维流线方法实现三维流场可视化[1]。流线指在流体中跟踪一组流动线上的流体质点,以展示流体的流动方向和速度分布。因此,流线可视化方法可以更好地表现流场的连续性。然而,在三维流场可视化中,三维流线之间的遮挡是主要难题之一[2],如果能够很好

到稿日期:2023-05-06　返修日期:2023-10-20
基金项目:国家自然科学基金(61702455,61672462);浙江省自然科学基金(LY20F020025)
This work was supported by the National Natural Science Foundation of China(61702455,61672462) and Natural Science Foundation of Zhejiang Province,China(LY20F020025).
通信作者:秦绪佳 (qxj@zjut.edu.cn)

地避免这些遮挡，就能得到更加真实的三维流场。

目前，解决这个问题有两种主要方法。第一种方法是通过流线特征提取和聚类，交互式地选择少量关键流线进行绘制，在保留关键信息的情况下减少流线之间的遮挡。这种方法通常需要对流场数据进行预处理，采用深度学习技术提取流线特征[3]，然后使用聚类算法选择关键流线进行绘制。第二种方法是在流线绘制时添加光照效果，使流线具有空间层次感。这种方法可以通过模拟光线在流线表面的反射和折射来模拟光照效果，从而使流线之间的遮挡效果减弱。

本文的主要内容分为以下两个部分：

1)使用深度学习进行流线特征提取。流线特征提取是流线可视化中的重要环节，对于复杂的流场数据，传统的流线提取方法往往难以快速准确地得到流线特征。而本文提出了一种基于卷积自编码器的新方法，其使用深度学习技术，可以自动提取复杂的流线特征。

2)使用改进 CFSFDP 算法的流线特征聚类。流线聚类是流线可视化中的另一个重要步骤。原 CFSFDP[4] 算法在聚类中心的选择过程中，只考虑聚类中心之间的距离不能太小，而忽略了选择流线时需要考虑流线之间的距离以尽可能覆盖整个矢量场的情况。因此，本文提出了一种改进的 CFSFDP 算法，该算法可以自适应地计算距离参数，并自动选择聚类中心进行聚类。

2 相关工作

目前选择关键流线的方法主要有两种。一种是种子点放置算法。该方法直接在流场中选择关键点作为种子点，然后跟踪关键流线。这种方法通常适用于已知关键点的情况，例如需要观察某个特定区域的流线分布情况。另一种是从大量生成的流线中提取特征。该方法通常用于需要观察整个流场的情况。流线特征可以包括其长度、方向、弯曲度等。然后，使用聚类方法去除相似的流线，减少需要绘制的流线数量，以降低流场特征的损失。这种方法的优点是可以全面地观察整个流场的特征，但需要耗费更多的计算资源。

最初，Turk 等[5]提出了一种图像引导的方法。该算法利用能量函数来刻画当前所选种子点与预期种子点在位置上的区别，从而优化种子点的摆放。该算法对流线进行创建、融合、再定位、扩展或者缩短，并对初始位置进行迭代细化，使能量函数达到最小。Chen 等[6]把种子点放置算法从二维流场推广至三维流场。这种算法生成的布局质量较高，但其过大的计算量制约了算法的实用性。Yusoff 等[7]在进行流场可视化时，首先对整个流场数据集进行了网格划分，以便于更加方便地处理和分析数据，然后选取了网格中速度最大和速度最小的点作为关键点，因为这些点在流场中通常代表着重要的流动特征和信息，最后采用了均匀散布的策略为其他区域选取种子点，以确保流线在整个流场中能够得到充分的覆盖和分布。Qin 等[8]在传统算法的基础上引入了信息熵，将其用于动态计算流线间隔，并根据流场不同区域的特征来布置种子点，以此来突出重要区域流线的特征，同时减少非重要区域流线的数量，提高计算效率。

对流场进行可视化时，选取具有代表性的流线已经成为种子点布置的一种可供选择的方案。Marchesin 等[9]对每条流线在流场中的作用进行了量化，并选择了对矢量场理解作用较大，同时导致视觉混乱可能性较小的流线。这种筛选方法旨在使可视化的结果更加清晰易懂。Ma 等[10]提出了一种以重要性驱动的流线选择方法，通过保证在视图逐渐变化的情况下流线的更新连贯性，使得可视化结果更具有连续性和可比性，以此帮助用户理解流场的演化过程，以及在不同视角下流线的分布情况。Qin 等[11]提出了一种方法，在保证流场特性的同时，引入均匀网格来控制流场的流线密度，以确保流场的均匀分布。Liu 等[12]使用主成分分析(PCA)方法来降低流线特征的维度，然后应用聚类方法对流场进行简化，将流场中具有不同特征的流线分开，提升了流场的可视化效果。

以上流线选择算法均基于人工定义特征进行选择，而目前卷积神经网络已经在计算机视觉领域证明了其强大的特征提取能力[13]。Han 等采用深度学习的方法，提出了一种基于卷积自编码器的流线和流面选择方法，称为 FlowNet[3]。这种方法的核心思想是，先将从矢量场中抽取出的流线和流面绘制成二值体，然后利用自编码器对二值体进行训练。接着使用训练好的编码器对二值体进行特征提取，并对结果使用 T-SNE 算法[14]进行降维处理，以便于它的可视化。最后，利用 DBSCAN 算法[15]对降维后的流线特征进行聚类。An 等[16]同样基于深度学习的方法，将低精度的流线输入网络中，然后对流线进行超精度提升，极大降低了流线的计算量，实现了流线的实时可视化。Lee 等[17]提出了一种深度回归模型，通过学习速度场中的流量趋势，来预测流场各处的流量重要性水平，并采用了曲线聚类策略进行流线选择。与之前的方法相比，深度学习的方法可能对设备的配置要求较高，但是它仍然为未来更精确地分析三维流场数据提供了新的思路和技术手段。

3 基于卷积自编码器的流线特征提取

由于三维矢量场的数据来源不同，有些数据来自于流体力学模拟，而另一些则来自于现实中的观测数据，其记录值的范围和完整性也不同。因此，在生成流线之前需要对这些三维矢量场数据进行处理。

3.1 三维流场的流线生成

本文使用了多个三维流场数据集，这些数据集的详细信息如表 1 所列。Tornado[3]数据集是一个由代码生成的龙卷风数据集；Cylinder[18]数据集是通过数值模拟得到的三维不可压缩流场数据集；Ctbl3d 数据集由德国气候计算中心和马克斯普朗克气象研究所提供，其使用了 UCLA-LES 模型，完成了对云尺度边界层的模拟；Tangaroa 数据集是对 Tangaroa 号科考船 CAD 模型周围的三维不可压缩流场的模拟，其使用了 Gerris 流求解器完成模拟，并将感兴趣区域重新采样到规则网格上；Func 数据集是根据 Han 等[19]所提的公式 $V(x,y,z)=[x,-y^2+1,z]$ 生成的。

表 1　数据集描述

Table 1　Description of datasets

数据集	维数	来源
Tornado	128×128×128	文献[3]
Cylinder	128×128×128	文献[18]
Ctbl3d	120×180×300	cgl. ethz. ch
Tangaroa	384×384×100	cgl. ethz. ch
Func	128×128×128	文献[19]

为了满足所需的数据尺寸，本文将流场的中心点坐标设置为[0,0,0]，并将整个流场映射至[−1,−1,−1]到[1,1,1]的范围内。这样的映射可以使流场数据在处理过程中保持一致的尺度，方便后续的特征提取和算法处理。

首先，需要对三维矢量场数据进行归一化的预处理。三维矢量场表示各网格点上的矢量，可以记为(v_u, v_v, v_w)。分别获取每个网格点p_t的矢量值的大小v_w和矢量的方向$d(p_t)$。接着求出所有网格点矢量值的最大值$v_{\max}$和最小值$v_{\min}$，利用式(1)对网格矢量值v_M进行归一化处理。每个网格点p_t上的矢量经过归一化，其值为v_m'，其方向仍为$d(p_t)$。

$$v_m' = \frac{v_M - v_{\min}}{v_{\max} - v_{\min}} \tag{1}$$

关于流场流线生成，有许多现有的方法可供使用。本文采用了 Yusoff 等[7]提出的方法。该方法将三维向量场划分成若干等大小的子区域，这样可以更有效地跟踪流线，并避免在整个向量场中计算流线的计算负担过大。然后，使用每个子区域中物理量的最大值和最小值作为流线跟踪的起点。这样的起点选择可以保证流线的起始点与向量场中的最显著的流线方向对应。随后的流线计算采用了四阶龙格库塔法，以确保流线的精度和稳定性，同时对其进行双向跟踪，这样可以避免由于流线被流场中的涡旋所吸引而无法到达目标区域，从而遗漏重要的流线特征的情况。通过以上步骤，我们可以获得流场的流线集合。

3.2　流线节点规则采样

在对流场进行流线提取之后，各节点所保存数据的维度相同，但是各流线上节点的个数未必一致。因为之后需要将流线输入神经网络进行提取特征，这就需要将输入数据的维度调整为一致。

在提取到流线后，由于其并非直线，因此本文采用非线性的 Hermit 插值的方法对流线进行重采样。这是一种基于局部样条插值的方法。在 Hermit 插值中，使用每个节点处的函数值和一阶导数值来计算插值曲线。这种方法在保持原有曲线形状的同时，能够更准确地描述曲线的局部特征，避免了线性插值导致的过度平滑和信息丢失的问题。重采样的过程如图 1 所示。

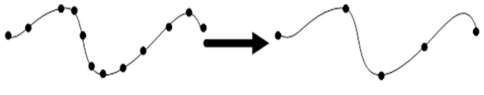


图 1　流线重采样

Fig. 1　Resample streamline

假设要将一条流线的节点数从 n 个调整为 m 个，使用$D(p_i, p_j)$表示节点p_i与节点p_j之间的距离，首先需要根据式(2)计算原始的 n 个节点到流线起始节点的路径长度τ_i($i=0,1,\cdots,n-1$)。

$$\tau_i = \begin{cases} 0, & i=0 \\ \tau_{i-1} + D(p_i, p_j), & i>0 \end{cases} \tag{2}$$

流线的长度即为最后一个节点到起始节点的距离，即τ_{n-1}，采样的步长 α 可由式(3)计算得到：

$$\alpha = \frac{\tau_{n-1}}{m-1} \tag{3}$$

用s_i($i=0,1,\cdots,m-1$ 且$s_0=p_0, s_{m-1}=p_{n-1}$)表示第 i 个采样点，τ_{s_i}表示第 i 个采样点到流线起始节点的距离。若要计算s_i的空间坐标，则需要先找到 j，使得$\tau_{s_i} \in [\tau_{j-1}, \tau_j]$，则$s_i$的空间坐标 $H(\mu)$可由式(4)计算得到：

$$H(\mu) = A\mu^3 + B\mu^2 + C\mu + D \tag{4}$$

其中，μ 表示点s_i到p_{j-1}与p_j的距离之比，如式(5)所示：

$$\mu = \frac{D(s_i, p_{j-1})}{D(p_{j-1}, p_j)}; \mu \in [0,1] \tag{5}$$

A, B, C, D 4 个系数的计算方法如式(6)所示：

$$\begin{aligned} A &= 2H(0) - 2H(1) + H'(0) + H'(1) \\ B &= -3H(0) + 3H(1) - 2H'(0) - H'(1) \\ C &= H'(0) \\ D &= H(0) \end{aligned} \tag{6}$$

边缘条件 $H(0), H(1), H'(0), H'(1)$的计算式如式(7)所示：

$$\begin{aligned} H(0) &= p_{j-1} \\ H(1) &= p_j \\ H'(0) &= D(p_{j-1}, p_j) v(p_{j-1}) \\ H'(1) &= D(p_{j-1}, p_j) v(p_j) \end{aligned} \tag{7}$$

其中，$v(p_j)$表示点p_j处的速度。

3.3　自动编码网络的设计与流线特征学习

本文方法使用卷积自编码器来对重采样后的流线进行特征提取。自编码器是一种无监督学习算法，它包括编码器和解码器两部分。编码器将输入的流线转换为一个低维向量，表示流线的关键特征。解码器将这个向量转换回原始流线的形式，以重建输入数据。通过训练，最终输出数据越接近于输入数据，就意味着自编码器学习到的数据特征更准确。

该网络的结构如图 2 所示。

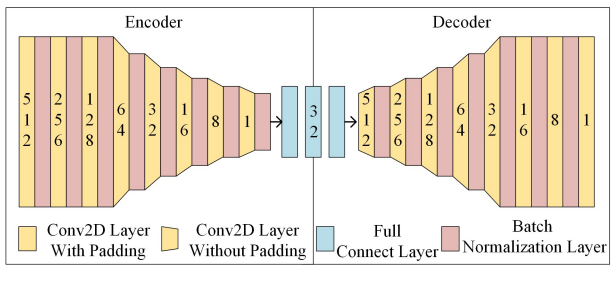


图 2　用于特征学习的编码器网络结构

Fig. 2　Structure of autoencoder for feature learning

其中，编码器由 8 层卷积层组成，用于提取非线性特征。首先使用 3 层带填充的卷积层来提取输入向量的特征。卷积核大小为(3,3)，带填充的卷积层可以防止输入向量在卷积过程中过度缩小，从而保留更多的信息。然后，使用 3 层不带

填充的卷积层来进一步提取输入向量的全局特征，卷积核大小仍然为(3,3)。其中输入向量的向量宽度为7，包括三维的空间位置、三维的速度方向以及速度的大小。经过前6层卷积后，向量的宽度将缩减为1。这意味着，每个元素将代表整个输入向量的特征。为了进一步压缩特征向量的大小，后续的第7层和第8层卷积层分别使用了大小为(4,1)和(3,1)的卷积核。这两个卷积层都不带填充。由于卷积层中卷积步长均为(1,1)，因此卷积过程中不会丢失太多信息。

解码器由2层全连接层和8层转置卷积层组成。其中，全连接层可以将特征向量的维度从解码器输出的大小扩大为编码器期望的输入维度，并且能捕获特征之间的复杂关系，从而进一步提高网络的性能。卷积层网络结构与编码器的卷积层互相对称，只是将编码器中的卷积层替换为转置卷积层。转置卷积层与卷积层相反，可以将特征向量的大小扩大。为了避免过拟合以及梯度消失的问题，解码器中使用了批归一化的处理方式。为了加快网络的训练过程并增强网络的非线性能力，网络的每一层的激活函数都被设置为ReLU函数，如式(8)所示：

$$\mathrm{ReLU}(x)=\begin{cases}x, & \text{if } x>0\\ 0, & \text{if } x\leqslant 0\end{cases} \tag{8}$$

其中，ReLU激活函数具有以下优点：

1)ReLU函数形式非常简单，其在输入值为正时，导数为1，在输入值为负时，导数为0。因此在反向传播中，ReLU函数的梯度计算可以很方便地进行，这使得神经网络的训练速度大大加快。

2)ReLU函数是非线性函数，可以让网络更好地拟合非线性问题。

3)ReLU是一个非饱和的激活函数，其导数在值过大或过小时不会接近于0，如同sigmoid或tanh函数那样，有效防止了梯度消失的问题。

4)ReLU函数将所有小于0的部分映射为0，而正值则保持不变。这种单向抑制使神经网络中的神经元被稀疏激活，可以有效地减少过度拟合现象。

在神经网络中，代价函数(也称为损失函数)用于度量模型预测的输出与真实值之间的差异。其作用是帮助优化算法找到最优的模型参数，以使得预测输出尽可能地接近真实值。常见的代价函数包括均方误差(MSE)、平均绝对误差(MAE)、交叉熵(Cross Entropy)和二值交叉熵(Binary Cross Entropy)等。通常情况下，在训练过程中，神经网络通过反向传播算法来计算代价函数对模型参数的梯度，然后利用如梯度下降等优化算法来更新模型参数，以最小化代价函数。由于本文涉及的是回归问题，因此只选择MSE或MAE作为代价函数，它们的计算式如式(9)、式(10)所示。

$$MSE(Y,\hat{Y})=\frac{1}{N}\sum_{i=0}^{N-1}(Y_i-\hat{Y}_i)^2 \tag{9}$$

$$MAE(Y,\hat{Y})=\frac{1}{N}\sum_{i=0}^{N-1}|Y_i-\hat{Y}_i| \tag{10}$$

在反向传播的过程中，优化函数用于更新模型的参数。使用不同的优化函数会影响模型的性能和收敛速度。常见的优化函数包括随机梯度下降(SGD)、自适应矩估计(ADAM)和牛顿动量加速自适应矩估计(NADAM)。SGD是一种最基本的优化算法，其思想是每次从训练集中随机抽取一个样本进行梯度计算和参数更新。由于SGD仅使用单个样本进行更新，因此可以加速收敛并节省内存。ADAM在SGD的基础上，通过计算梯度的一阶和二阶矩估计来自适应地调整学习率。NADAM是在ADAM的基础上加入了Nesterov动量项的一种优化算法，Nesterov动量项可以加速收敛并增加稳定性，同时减小了ADAM算法可能导致的过拟合问题。然而，近年来有研究[20]表明，ADAM和NADAM可能不收敛并错过全局最优解。在本文的多组比较实验中，我们选择了SGD作为自编码器的优化函数，并采用MSE作为代价函数。

4 改进的CFSFDP算法的流线聚类

流线特征提取完成后，为了便于结果的可视化显示，首先减少从流线中提取的特征的维度，然后使用改进的CFSFDP算法进行聚类。参考FlowNet的选择，本文使用T-SNE算法对自编码器提取特征进行降维。CFSFDP算法需要保证聚类中心具有相对较大的密度并且远离其他聚类中心。首先，根据截断距离参数d_c，计算每个数据点的密度。然后，按密度从大到小将数据点排序，并计算每个数据点到最近具有较高密度的数据点的距离δ，生成决策图。用户根据决策图选择聚类中心。最后，基于密度的大小为其他非中心点进行分类。有关CFSFDP算法的更详细描述请参考文献[4]。本文弥补了原始CFSFDP算法的两个缺陷，即对距离参数敏感以及需要手动选择聚类中心。

4.1 改进的CFSFDP算法

首先为了解决原始CFSFDP算法的距离参数敏感性问题，本文引入了非参数密度估计方法。与传统方法相比，非参数方法更加灵活，不会受到特定偏差的影响。核密度估计是最常用的非参数密度估计方法，对于数据点$\{x_0,x_1,x_2,\cdots,x_{n-1}\}$，核密度估计将它们描述为概率密度函数，如式(11)所示：

$$\hat{f}_h(x)=\frac{1}{n}\sum_{i=0}^{n-1}K(x-x_i) \tag{11}$$

其中，K表示核函数，n表示数据集中的样本数。在核密度估计中，高斯核函数是最常用的核函数之一。高斯核函数如式(12)所示，其中h表示核函数的带宽。带宽的选择在核密度估计中十分重要，如果带宽过大，则估计结果会过于平滑，忽略了数据中的细节信息；而如果带宽过小，则估计结果会过于敏感，忽略了数据的整体分布趋势。

$$K(x,x_i,h)=\frac{1}{\sqrt{2\pi h}}\mathrm{e}^{-\frac{(x-x_i)^2}{2h}} \tag{12}$$

积分平方误差(MISE)是一个用于确定带宽h最佳值的指标，计算式如式(13)所示：

$$\begin{aligned}MISE(h)&=E\left[\int(\hat{f}_h(x)-f(x))^2\mathrm{d}x\right]\\&=AMISE(h)+o\left(\frac{1}{nh}+h^4\right)\end{aligned} \tag{13}$$

其中，$\hat{f}_h(x)$表示估计密度；$f(x)$表示实际密度。

MISE的值越小，表示核密度估计的结果越优，其中

AMISE 的计算式如下：

$$R(g)=\int g(x)^2\mathrm{d}x$$
$$m_2(K)=\int x^2K(x)\mathrm{d}x \tag{14}$$
$$AMISE(h)=\frac{R(K)}{nh}+\frac{1}{4}m_2(K)^2h^4+R(f'')$$

根据式(13)，要得到 MISE 的最小值，则需使 AMISE 最小，因此可以将 AMISE 对变量 h 求偏导，计算式为：

$$\frac{\partial AMISE(h)}{\partial h}=-\frac{R(K)}{nh^2}+m_2(K)^2h^3R(f'') \tag{15}$$

最后求解式(15)得：

$$h_{\mathrm{AMISE}}=\frac{R(K)^{\frac{1}{5}}}{m_2(K)^{\frac{2}{5}}R(f'')^{\frac{1}{5}}n^{\frac{1}{5}}} \tag{16}$$

当数据为正态分布时，最优的 h 可由式(17)得到，其中 n 为样本的个数，σ 为标准差。

$$h=1.06\sigma\times n^{-0.2} \tag{17}$$

Sun 等[21]使用 Solve-the-Equation 方法对 $R(f'')$ 进行优化，将式(16)进一步推导为：

$$k_1=((x-x_i)^2-6h^2)^2-24h^4$$
$$k_2=\mathrm{e}^{-\frac{(x_i-x_j)^2}{2h}}$$
$$\hat{h}_{\mathrm{opt}}=\left\{\frac{4nh^6}{3nh+\dfrac{1}{2h^3\sum\limits_{i=0}^{n-1}\sum\limits_{j<i}k_1k_2}}\right\}^{\frac{1}{5}} \tag{18}$$

式(18)是计算 h 的最佳值的迭代过程，方程中的符号含义与式(11)中的相同。本文参照 Sun 等[21]的方法，将式(17)得到的结果设为 h 的初始值，可以加快计算速度并减少迭代次数。

若数据的维数为 d，对每一个维度都计算其 h 值，第 i 个维度的 h 值记为 h_{i-1}，通过式(19)计算最终聚类算法的截断距离 d_c。

$$d_c=\sqrt{\sum_{i=0}^{d-1}h_i^2} \tag{19}$$

经过上述步骤，本文基于非参数密度估计的方法，实现了对聚类过程中的截断距离参数的自适应计算，解决了原始 CFSFDP 算法对截断距离参数敏感的问题。下文将说明如何实现聚类中心的自动选取。

根据原始 CFSFDP 算法的描述，CFSFDP 的聚类中心需要有较大的密度，并且距离密度高的数据点较远。使用 ρ_i 表示数据点 x_i 的密度，δ_i 表示数据点 x_i 到密度更高的点的最短距离，ρ_i 和 δ_i 的计算式如式(20)和式(21)所示：

$$\rho_i=\sum_j\chi(d_{ij}-d_c)$$
$$\chi(x)=\begin{cases}1, & x<0\\ 0, & \text{otherwise}\end{cases} \tag{20}$$
$$\delta_i=\begin{cases}\min\limits_{j:\rho_j>\rho_i}(d_{ij}), & \text{if } \rho_i<\max\{\rho_j\}\\ \max\limits_j(d_{ij}), & \text{if } \rho_i=\max\{\rho_j\}\end{cases} \tag{21}$$

其中，d_{ij} 表示数据集中数据点 i 和数据点 j 的欧几里得距离，d_c 表示算法的截断距离参数。

为了便于判断，使用一个指标 γ 代替上述两个指标，γ 的计算式如式(22)所示。由于计算出来的 ρ_i 和 δ_i 的值域可能存在差距，因此值域大的指标会降低另外一个指标的影响力，并对结果产生较大影响。因此，在计算最终指标前，需要对计算得到的 ρ 和 δ 进行归一化，其方法与预处理流线时使用的方法相同，计算式如式(1)所示。

$$\gamma_i=\delta_i\rho_i \tag{22}$$

接着，计算所有数据点对应的 γ 值，并根据 γ 值从大到小对数据点进行排序。根据输入的聚类量 K，从前到后选择 K 个数据点作为聚类中心，这样就实现了聚类中心的自动选取，避免了原始 CFSFDP 算法中聚类中心的手动选择过程。最后，根据 CFSFDP 方法对其他数据点进行分类。

改进的 CFSFDP 算法的具体步骤如算法 1 所示。

算法 1　改进的 CFSFDP 算法步骤

输入：所有数据点

输出：若干个聚类中心和聚类完成的数据点

Step1　根据式(17)计算 h_0 的初始值，并将 h_0 代入式(18)计算得到 h_1

Step2　如果 $|h_0-h_1|$ 小于阈值 ε，则转到步骤 4；否则进入步骤 3

Step3　将 h_1 赋值给 h_0，将 h_0 代入式(18)计算得到新的 h_1，并将 h_1 设为 h_0 与 h_1 的中间值，返回步骤 2

Step4　令 $\hat{h}_{\mathrm{opt}}=h_1$，并利用式(19)和计算出的 $\{h_0,h_1,h_2,\cdots,h_{d-1}\}$ 完成对截断距离 d_c 的计算

Step5　根据截断距离 d_c，按照 CFSFDP 算法步骤，计算每一个数据点 x_i 对应的 ρ' 和 δ'

Step6　对 ρ 和 δ 进行归一化得到 ρ' 和 δ'，并将结果代入式(20)完成对 γ 的计算

Step7　将数据点按照 γ 从大到小进行排序，根据输入的聚类个数 K，选取排序后的前 K 个数据点作为聚类中心

Step8　按照 CFSFDP 的步骤为其他作为非聚类中心的数据点划分类别

4.2　流线特征降维与流线聚类

本文中，自编码器提取的流线特征维度达到了 32 维，这可能会降低聚类算法的性能，并且高维数据的可视化相对困难，因此对流线特征进行了降维处理。在现有的降维算法中，T-SNE(t-Distributed Stochastic Neighbor Embedding)[14] 是最优秀的一种。T-SNE 算法的主要思想是将高维数据映射到低维空间(通常是二维或三维空间)，使得相似的数据在低维空间中距离较近，不相似的数据距离较远。在这个过程中，T-SNE 使用概率分布的方式来描述数据点在高维空间和低维空间中的分布，同时通过最小化高维空间和低维空间的距离来找到合适的映射方式。与其他降维算法相比，T-SNE 的优点是可以保留数据的局部结构信息，并且在可视化高维数据时具有很好的效果。

本文方法在应用 T-SNE 算法后，流线特征被降到 2 维，之后再应用改进的 CFSFDP 算法，对降维后的数据进行聚类，并选取了若干个聚类中心。接着利用这几个聚类中心所代表的流线来重构矢量场。在此过程中，选择流线时应更加倾向于展示矢量场的全局特征，至于矢量场的局部特征，则可以通过绘制不同类别的流线来展示。这种选择方式能够更好地呈现矢量场的整体结构，有助于更深入地理解流场数据。一些聚类结果如图 3 所示，这些结果是不同簇中的流线，展示了不同矢量场的局部特征。

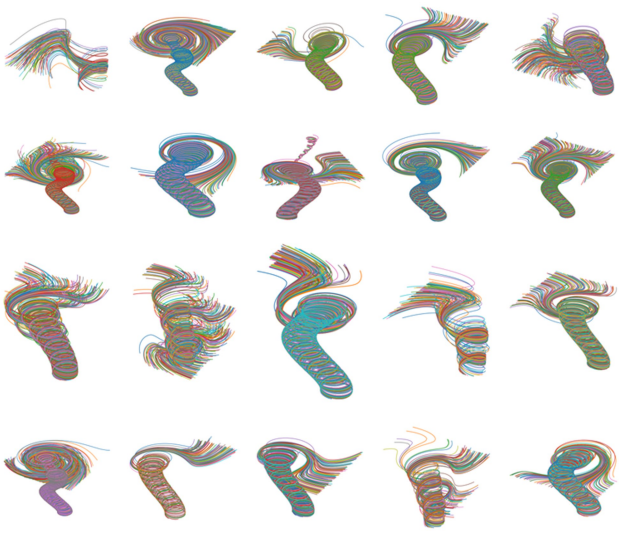


图 3 不同簇的流线

Fig. 3 Streamlines of different clusters

4.3 改进的聚类算法的性能分析

上述改进后的 CFSFDP 算法只需对数据点进行一次遍历就能够完成特征聚类，但在数据集中样本数量较大时，聚类仍需花费较长时间。从表 2 中的数据可以看出，数据集中的样本数量越多，聚类所需时间就越长，并且对距离矩阵和截断距离参数 d_c 的计算占用了大量时间。而改进后的 CFSFDP 算法的一个优点是，计算过程中的距离矩阵和截断距离参数 d_c 不会随聚类数量改变。

表 2 聚类算法的时间消耗

Table 2 Time consumption of clustering algorithms

数据集	样本数	总时间/s	计算截断距离的时间/s	计算距离矩阵的时间/s
Tornado	6974	464.12	456.73	3.94
Cylinder	11655	929.16	909.07	11.16
Ctbl3d	8435	646.97	636.54	6.21
Tangaroa	13559	1073.88	1052.61	11.82
Func	7062	383.45	375.76	4.50

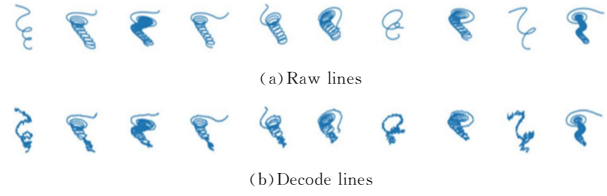
由算法 1 可知，每次改变聚类数量只需要在改进后的 CFSFDP 算法中再进行步骤 7 和步骤 8。因此，在绘制前计算所有节点的 γ 值，计算方法如式(20)所示。然后，根据 γ 值从大到小对节点进行排序。同时，为了快速分类非聚类中心数据点，本文采用了一种优化策略。具体来说，对于每个节点，算法会寻找距离它最近并且密度大于它的节点，并且记录其在原始数据中的类别。这样，在分类时就可以快速定位到该数据点的局部密度，并且可以通过该节点的类别来确定该数据点的距离关系，从而加快分类的速度。此外，由于聚类结果需要按照原始数据的顺序传输到 GPU，因此可以通过记录每个节点在原始数据中的索引来避免重新排序，提高算法效率。

5 实验结果与分析

将从特征学习的结果与流线聚类的结果两个方面，将本文方法与其他不同方法进行比较。本文实验使用的三维流场数据集如表 1 所列。

5.1 特征学习结果分析

图 4 给出了训练好的自编码器的输入和输出之间的对比。图 4 中，第一行显示了原始的流线，第二行显示了训练后的自编码器输出的结果。从图中可以看出，自编码器输出的结果与原始流线的整体形状基本一致，这说明自编码器成功提取了流线特征。此外，将比较本文方法与其他现有的流线选择方法和种子点放置方法，分别是 Han[3]，Xu[22] 和 Yusoff[7] 方法。本文和 Han 的 FlowNet 方法都是基于学习的方法，而 Xu 和 Yusoff 的方法是人工设计的特征方法。



(a) Raw lines

(b) Decode lines

图 4 流线解码结果

Fig. 4 Results of streamline decode

利用以上 3 种方法和本文方法所选择的流线分别对矢量场进行重构，并对重构后的矢量场和原始场做了对比。本文采用 PSNR 与 AAD 两个指标判断两种流场的相似度。

PSNR(Peak Signal-to-Noise Ratio)指峰值信噪比，通过计算原始流场与重建流场之间的峰值信噪比，来衡量重建流场与原始流场之间的差异程度。AAD(Average Absolute Difference)指平均角度差，通过计算原始流场与重建流场之间的平均绝对差异，来衡量重建流场与原始流场之间的平均误差。两者的计算式分别如式(23)和式(24)所示：

$$PSNR(V,V')=20\log_{10}I(V)-10\log_{10}MSE(V,V') \tag{23}$$

其中，V 为原矢量场，V' 为重构矢量场，$I(V)$ 是原矢量场中最大值与最小值之差，$MSE(V,V')$ 是原矢量场与重构矢量场之间的均方误差。

$$AAD(V,V')=\frac{\sum_{i=0}^{n-1}acos(dot(norm(V_i),norm(V'_i)))}{n\pi} \tag{24}$$

其中，V_i 表示原矢量场 V 中位置 i 的数据，V'_i 则表示重构矢量场 V' 中位置 i 的数据，$norm$ 即归一化操作，dot 即点乘操作。从数值上来看，PSNR 值越大，或者 AAD 值越小，说明这两个矢量场越接近。实验计算结果如表 3、表 4 所列。

表 3 不同关键流线选取方法的 PSNR

Table 3 PSNR of different key streamline selection methods

数据集	选择的流线数量	Xu 的方法	Yusoff 的方法	FlowNet	本文方法
Tornado	96	17.242	17.677	17.88	18.209
Cylinder	133	4.778	6.532	6.641	6.599
Ctbl3d	201	24.634	25.176	25.301	25.762
Tangaroa	165	4.329	10.676	11.86	13.293
Func	159	7.642	10.666	15.751	16.948

表 4 不同关键流线选取方法的 AAD

Table 4 AAD of different key streamline selection methods

数据集	选择的流线数量	Xu 的方法	Yusoff 的方法	FlowNet	本文方法
Tornado	96	0.2400	0.2370	0.2450	0.2310
Cylinder	133	0.4850	0.2540	0.2540	0.2530
Ctbl3d	201	0.5570	0.5570	0.5570	0.5570
Tangaroa	165	0.4010	0.0210	0.0019	0.0021
Func	159	0.3090	0.0747	0.0694	0.0692

表 5　FlowNet 和本文方法的运行时间对比

Table 5　Comparison of runtime between FlowNet and our method

数据集	流线数量	每次迭代所需秒数	
		FlowNet	本文方法
Tornado	6974	3375	145
Cylinder	11655	5669	214
Ctbl3d	8435	3876	145
Tangaroa	13559	6562	226
Func	7062	3420	139

由表 3、表 4 可见,将本文方法和 FlowNet 这两种以学习为基础的方法与比他两种非学习方法对比时,基于学习的方法在结果上的准确性比非学习方法高。并且同样作为基于学习的方法,本文方法的结果略优于 FlowNet。同时,表 5 列出了在不同数据集上使用本文方法和 FlowNet 迭代一次所需的时间对比结果,显然本文方法的计算效率远远优于 FlowNet。

5.2　流线聚类结果分析

使用 T-SNE 算法来降低自编码器输出的流线特性的维数,然后使用改进的 CFSFDP 算法进行聚类,并选择聚类中心的流线作为关键流线来重建矢量场。此外,本文选择了其他 4 种不同方法,通过选择相同数量的流线来重建矢量场与本文方法进行比较。与 5.1 节相同,采用 PSNR 与 AAD 两个指标判断流场相似度,比较结果如表 6、表 7 所列,表中的 AE 表示本文中的自编码器。

表 6　不同关键流线选取方法的 PSNR

Table 6　PSNR of different key streamline selection methods

数据集	流线数量	Xu 的方法	Yusoff 的方法	FlowNet	AE+DBSCAN	本文方法
Tornado	40	15.534	15.565	17.216	17.097	16.716
Cylinder	69	4.773	5.266	6.292	6.286	6.286
Ctbl3d	201	24.634	25.176	25.301	25.762	25.248
Tangaroa	165	4.329	10.676	11.862	13.293	13.959
Func	215	7.668	11.030	17.475	18.653	19.536

表 7　不同关键流线选取方法的 AAD

Table 7　AAD of different key streamline selection methods

数据集	流线数量	Xu 的方法	Yusoff 的方法	FlowNet	AE+DBSCAN	本文方法
Tornado	40	0.267	0.267	0.2550	0.256	0.2450
Cylinder	69	0.492	0.380	0.2780	0.274	0.2720
Tangaroa	165	0.401	0.021	0.0194	0.021	0.0197
Func	215	0.304	0.071	0.0690	0.068	0.0680

从图 5 可以看出,使用学习方法的最后三列结果比使用非学习方法的前两列结果好。在一些数据集中,如龙卷风数据集 Tornado,其流场中心的流线比周围的流线更密集,而 CFSFDP 算法要保证聚类中心之间的距离不能过小,这导致其重建矢量场时不能涵盖到流场中心的大量流线。在这种情况下,使用 DBSCAN 聚类算法则能够更好地选择流线来重建矢量场,因为它会选择更多位于流场中心的流线。但对于其他数据集,例如 Func,流线分布更加均匀,虽然 CFSFDP 算法选择的流线数量比 DBSCAN 聚类算法更少,但是在这种情况下,所选的流线足以涵盖整个流场,从而更好地重建矢量场。

图 5 给出了使用不同算法提取的流线,可以看到,本文方法提取的流线间隙比其他两张使用学习方法的大,这也证实了上述说法。

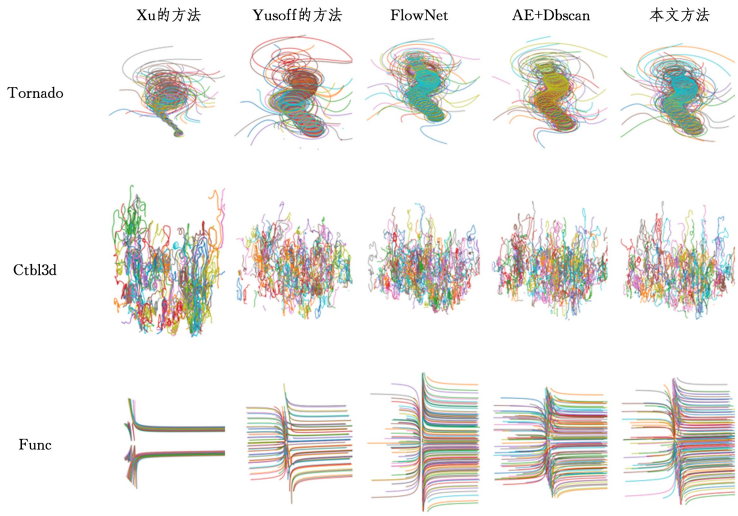


图 5　流线聚类结果

Fig. 5　Streamline clustering results

结束语　为了更好地观察和分析流场的运动变化和规律,需要提取流场的特征,并从不同角度观察流场的特征。流线是流场特征的重要表现形式,也是流场特征可视化的一种重要方法。为了快速方便地选择代表某一流场特征的流线

聚类，消除三维流线之间的相互遮挡，使用户能够准确地观察流场的特征，本文主要研究了三维流场的流线特征学习、聚类。本文的主要贡献如下：1)设计了一种新的自编码器网络，通过深度学习的方法获取流线的特征；2)提出了一种改进的CFSFDP算法，并对从流线中提取的特征进行聚类。

流面也是流场可视化的一种重要形式。本文方法不能完成流面的特征学习和选择，未来将对其进行进一步改进。

参 考 文 献

[1] SHAO X Q, CHENG Y, JIN Y Z. A review of the application of illustrative methods in 3D streamline visualization [J]. Journal of Graphics, 2022, 43(5): 753-764.

[2] ZHANG Q, XIAO L. Review of Visualization Drawing Methods of Flow Field Based on Streamlines [J]. Computer Science, 2021, 48(12): 1-7.

[3] HAN J, TAO J, WANG C. FlowNet: A Deep Learning Framework for Clustering and Selection of Streamlines and Stream Surfaces[J]. IEEE Transactions on Visualization and Computer Graphics, 2020, 26(4): 1732-1744.

[4] RODRIGUEZ A, LAIO A. Clustering by fast search and find of density peaks[J]. Science (American Association for the Advancement of Science), 2014, 344(6191): 1492-1496.

[5] TURK G, BANKS D. Image-guided streamline placement[C]// SIGGRAPH '96. ACM, 1996, 453-460.

[6] CHEN Y, COHEN J, KROLIK J. Similarity-Guided Streamline Placement with Error Evaluation[J]. IEEE Transactions on Visualization and Computer Graphics, 2007, 13(6): 1448-1455.

[7] YUSOFF Y A, MOHAMED F, MOKHTAR M K, et al. Magnitude-based seed point placement for streamlines generation [C]// 2017 IEEE Conference on Big Data and Analytics (ICBDA). 2018, 2018: 81-86.

[8] QIN X J, CHENG Y X, ZUO S H, et al. Streamline Placement Algorithm with Dynamic Spacing in Flow Field Controlled by Information Entropy [J]. Journal of Chinese Computer System, 2021, 42(12): 2632-2636.

[9] MARCHESIN S, CHEN C K, HO C, et al. View-Dependent Streamlines for 3D Vector Fields[J]. IEEE Transactions on Visualization & Computer Graphics, 2010, 16(6): 1578-1586.

[10] MA J, WANG C, SHENE C K. Coherent view-dependent streamline selection for importance-driven flow visualization[C]// Proceedings of SPIE. The International Society for Optical Engineering, 2013, 8654: 1-15.

[11] QIN X, CHEN X, CHEN L, et al. Streamline Uniform Placement Algorithm With Dynamic Seed Points[J]. IEEE Access, 2019, 7: 113844-113852.

[12] LIU F, ZHOU W, LIU B, et al. Flow Field Description and Simplification Based on Principal Component Analysis Downscaling and Clustering Algorithms[J]. Frontiers in Earth Science, 2022, 9: 804617.

[13] JIE H, LI S, GANG S, et al. Squeeze-and-Excitation Networks [J]. IEEE Transactions onPattern Analysis and Machine Intelligence, 2020, 42(8): 2011-2023.

[14] LAURENS V D M, HINTON G. Visualizing Data using t-SNE [J]. Journal of Machine Learning Research, 2008, 9 (2605): 2579-2605.

[15] SCHUBERT E, SANDER J, ESTER M, et al. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN[J]. ACM Transactions on Database Systems, 2017, 42(3): 1-21.

[16] AN Y F, SHAN G H, LI G, et al. Interactive Flow Visualization Based on Super-Resolution Streamline Generation[J]. Computer Systems & Applications, 2021, 30(10): 48-58.

[17] LEE J Y, PARK J. Deep regression network-assisted efficient streamline generation method[J]. IEEE Access, 2021, 9: 111704-111717.

[18] ZHANG W, PEI Y, LIU B, et al. A Streamline Illumination Method for 3D Flow Fields[C]// 2013 Seventh International Conference on Image and Graphics. 2013: 252-257.

[19] HAN J, TAO J, ZHENG H, et al. Flow Field Reduction Via Reconstructing Vector DataFrom 3-D Streamlines Using Deep Learning[J]. IEEE Computer Graphics and Applications, 2019, 39(4): 54-64.

[20] WILSON A C, ROELOFS R, STERN M, et al. The marginal value of adaptive gradient methods in machine learning[C]// Advances in Neural Information Processing Systems, 2017. 2017: 4151-4161.

[21] SUN L, JIANG M, GE Y. On theAdaptive Selection of the Cutoff Distance and Cluster Centers Based on CFSFDP[C]// Proceedings of the 38th Chinese Control Conference. 2019: 165-169.

[22] XU L, LEE T Y, SHEN H W. An Information-Theoretic Framework for Flow Visualization[J]. IEEE Transactions on Visualization and Computer Graphics, 2010, 16(6): 1216-1224.

CHEN Jie, born in 1998, master. His main search interests include computer graphics and digital image processing.

QIN Xujia, born in 1968, Ph.D, professor. His main research interests include computer graphics, image processing and data visualization.

（责任编辑：喻藜）