计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200500156

基于 TPE 的 SpaRC 算法超参数优化方法

邓 丽 武金达 李科学 卢亚康
上海大学机电工程与自动化学院 上海 200072
上海市电站自动化技术重点实验室 上海 200072

摘 要 宏基因组序列组装在计算和内存上面临着巨大挑战。SpaRC(Spark Reads Clustering)是基于 Apache Spark 的宏基因组序列片段聚类算法,为来自下一代测序技术的数十亿测序片段聚类提供了一种可扩展的解决方案。但是,SpaRC 算法参数的设置是一项非常具有挑战性的工作。SpaRC 算法拥有许多对算法性能有着很大影响的超参数,选择合适的超参数集对于充分发挥 SpaRC 算法的性能来说是至关重要的。为了提高 SpaRC 算法的性能,探索了一种基于树状结构 Parzen 估计方法(Tree Parzen Estimator,TPE)的超参数优化方法,其能够利用先验知识高效地调节参数,并通过减少计算任务加速寻找最优参数,达到最佳聚类效果,从而避免昂贵的参数探索。对长序列片段(PacBio)和短序列片段(CAMI2)进行实验,结果表明,该方法在改善 SpaRC 算法性能方面有着良好的效果。

关键词:SpaRC;宏基因组;序列片段聚类;TPE;超参数优化
中图法分类号 TP399;Q812

SpaRC Algorithm Hyperparameter Optimization Methodology Based on TPE

DENG Li,WU Jin-da,LI Ke-xue and LU Ya-kang
School of Mechatronic Engineering and Automation,Shanghai University,Shanghai 200072,China
Shanghai Key Laboratory of Power Station Automation Technology,Shanghai 200072,China

Abstract The assembly of metagenomic sequences faces huge challenge in computing and storage. SpaRC (Spark Reads Clustering) is a metagenomic sequence fragment clustering algorithm based on Apache Spark, which provides a scalable solution for clustering of billions of sequencing fragments. However, setting SpaRC parameters is a very challenging task. SpaRC algorithm has many hyperparameters that have a great impact on the performance of the algorithm. Choosing the appropriate hyperparameter set is crucial to the performance of SpaRC algorithm. In order to improve the performance of SpaRC algorithm, a hyperparameter optimization method based on Tree Parzen Estimator (TPE) is explored, which can use prior knowledge to efficiently adjust the parameters, accelerate the search for the optimal parameters by reducing the calculation task to achieve the optimal clustering effect, thus avoding expensive parameter exploration. After experiments with long-reads (PacBio) and short-reads (CAMI2), the results show that the proposed method has a great effect on improving the performance of SpaRC algorithm.

Keywords SpaRC, Metagenomics, Sequence fragment clustering, TPE, Hyperparametric optimization

1 引言

宏基因组学技术的兴起有效地解决了传统微生物学在检测和鉴定微生物物种方面存在局限性的问题。使用下一代测序(Next Generation Sequencing,NGS)[1]技术的全基因组鸟枪测序(Whole Genome Shotgun,WGS)是用于大型真核生物转录组分析和复杂微生物群落分析的强大工具[2]。由于 WGS 采用随机采样的方式且存在测序错误,测序片段必须覆盖单个基因或基因组很多次,以确保高质量的组装[3]。但是,在不了解物种结构的情况下很难精确估计所需的测序深度,对大型转录组和复杂元基因组进行测序通常会产生尽可能多的序列数据[4]。整体组装这些数据集需要更高效且可扩展的方法。目前常用的方法是使用多个线程(MEGAHIT)[5]或者大型集群(Message Passing Interface,MPI)。但是,这些试图从整体上组装宏基因组数据的方法很难随着数据量的增加实现近线性的扩展。

DIME(Divide conquer and MErge strategies)[6]首先使用了先分类再按类组装的方法,能够有效减少整体组装的数据量。但是该方法基于 Apache Hadoop 实现,Hadoop 的计算效率非常低,使其难以实现近线性的扩展。相比 Hadoop,

到稿日期:2020-05-29 返修日期:2020-12-05 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61802246)
This work was supported by the National Natural Science Foundation of China(61802246).
通信作者:邓丽(dengli@shu.edu.cn)

Apache Spark 在交互式计算和迭代计算方面具有非常大的优势。SpaRC（Spark Reads Clustering）是一种基于 Apache Spark 的宏基因组序列片段聚类算法[7]，其在组装之前先对序列片段进行聚类，以提高宏基因组序列组装算法的性能，当序列片段大量增加时可以实现近线性扩展。SpaRC 算法包含一些对聚类结果和算法性能影响较大的参数，这些参数的设置都涉及一定的专业知识，设置合适的参数才能发挥算法的最优性能。因此，SpaRC 参数的设置通常非常耗时，而且浪费资源。

目前，常用的超参数优化算法主要分为 3 类：网格搜索（Grid Search）、随机搜索（Random Search）和贝叶斯优化（Bayesian Optimization）。网格搜索通常搜索整个超参数空间，在高维空间容易遭遇维度灾难，是一种昂贵的搜索算法[8]；随机搜索的各组实验之间相互独立，不能利用先验知识选择下一组参数，容易漏掉搜索空间中一些重要的参数；贝叶斯优化是基于模型的超参数优化，能充分利用已测参数的信息，并根据历史信息决定下一个采样点，自适应地迭代计算[9]。其能够利用先验知识高效地调节参数，并通过减少计算任务来加速寻找最优参数的进程，不易陷入局部最优。基于顺序模型的优化方法（Sequential Model-sed Global Optimization，SMBO）是贝叶斯优化的形式化，而且其使用 TPE[10] 作为代理函数能够在相同数量的实验中找到比随机搜索更好的超参数。目前，尽管贝叶斯优化算法在某些机器学习领域获得了成功，但是在生物信息处理领域的应用仍然较少。

综上，本文提出了一种使用 TPE 作为代理函数的贝叶斯优化策略，用于提高 SpaRC 算法的聚类性能。

2　SpaRC 算法

2.1　算法概述

SpaRC 是一种基于 Apache Spark 的可扩展宏基因组序列片段的聚类算法，通过共享 k-mer 数量对序列片段（reads）进行分类，以实现下游组装的优化。SpaRC 通过序列片段测序技术能够在转录组和元基因组上产生很高的聚类性能。SpaRC 算法主要通过以下 3 个方面来实现近线性扩展：

1）通过共享 k-mer 数量估计成对序列的相似性（边缘权重）；

2）通过下采样来控制随着数据规模和复杂度的增长，由丰富的物种和噪声引起的数据爆炸；

3）应用快速重叠社区检测算法，即标签传播算法（Lable Propagation Algorithm，LPA）[11] 有效划分序列片段的图，并根据共享遗传信息来中断包含不同物种混合物的划分。

SpaRC 能够通过不同的参数设置对长序列片段（long reads）和短序列片段（short reads）进行聚类。SpaRC 首先计算一对序列片段（reads）之间的共享 k-mer 数，以近似其重叠数，然后构建无向序列片段（reads）图，最后进行图分区以形成簇。

具体来说，SpaRC 包含 5 个模块，主要是为长序列片段（LocalClustering）和短序列片段（GlobalClustering）而设计，分别是：KmerMapReads（MinimizerMapReads），GraphGen，GraphLPA，GlobalCluster 和 CCAddSeq。其中，KmerMapReads（MinimizerMapReads）把给定的序列片段根据预定义的序列长度将其切分为 k-mer，并且每条序列片段仅保留不同的 k-mer；GraphGen 根据 KmerMapReads（MinimizerMapReads）生成的 k-mer 来判断两个序列片段之间的 k-mer 重叠数，并根据两个序列片段之间重叠的 k-mer 数生成图；GraphLPA 对 GraphGen 生成的图进行分类；在上述模块中，序列片段用数字 ID 表示，以节省内存和存储空间，形成簇后，CCAddSeq 将检索序列片段并将其格式化，用于下游并行组装过程；GlobalCluster 使用一组短的、可靠的 k-mer 来估计每个簇丰度，然后计算簇之间的相似性，并用其来构建聚类图。

LocalClustering 主要包括 KmerMapReads，GraphGen，GraphLPA，CCAddSeq 4 个模块，用于长序列片段聚类；GlobalClustering 主要包括 MinimizerMapReads，GraphGen，GraphLPA，GlobalCluster 和 CCAddSeq 5 个模块，用于短序列片段聚类。

2.2　超参数集合

SpaRC 包含对聚类结果以及算法性能影响较大的参数。表 1 为 SpaRC 算法中部分参数设置对聚类指标的影响，其中，LocalCustering 采用纯度中位数、完整度中位数、纯度百分比以及完整度百分比四者的平均值作为最终评价指标；GlobalClustering 采用纯度中位数、完整度中位数二者的平均值作为最终评价指标。可以发现，*min_kmer_count*，*max_shanred_kmers*，*max_degree*，*kmercount* 参数的设置对 SpaRC 算法指标的影响不大。LocalClustering 部分的 *min_reads_per_cluster* 参数和 GlobalClustering 部分的 *size* 参数由用户根据需要进行设置，以控制形成簇的大小，对聚类性能并不会产生影响。

表 1　不同参数的影响

Table 1　Influence of different parameters

参数	取值	指标
min_kmer_count	5	68.77
	10	68.88
	15	68.86
max_shanred_kmers	5 000	76.745
	10 000	76.803
max_degree	25	76.75
	50	76.595
kmercount	9	37.17
	50	38.603
	100	37.243

为了节省计算资源，提高优化效率，选取 *k*，*min_shared_kmers*，*max_kmer_count*，*cosine* 4 个参数进行优化，其余没有优化价值的参数选择 SpaRC 官方文档推荐的默认值。

k 可以控制序列片段（reads）重叠的灵敏度和特异性；*max_kmer_count* 能够控制形成簇的大小，进而控制噪声污染；*min_shared_kmers* 用来判断两个测序片段之间的相似性；*cosine* 可以根据两个簇之间的相似性来判断再聚类簇的数量。通过实验发现这些都是对 SpaRC 运行结果影响较大的参数，在一定区间内适当调整这些参数可以极大地提高 SpaRC 的性能。这里根据 SpaRC 官方文档推荐的参数取值

范围对优化模型的参数取值范围进行设置。其中，k 的取值范围为(15，50)，max_kmer_count 的取值范围为(1000，4000)，min_shared_kmers 的取值范围为(2，10)，$cosine$ 的取值范围为(0.85，1.00)。其他参数使用 SpaRC 官方文档推荐参数，如表 2 和表 3 所列。

表 2 LocalClustering 超参数集

Table 2 LocalClustering hyperparameter set

模块	参数	区间
Kmer-MapReads	k	15～50
	min_kmer_count	2
	max_kmer_count	1000～4000
GraphGen	min_shared_kmers	2～10
	max_degree	50
GraphLPA	min_shared_kmers	2～10
	max_shared_kmers	20000
	$min\_reads\_per\_cluster$	2

表 3 GlobalClustering 超参数集

Table 3 GlobalClustering hyperparameter set

模块	参数	区间
Minimizer-MapReads	k	41
	m	21
	min_kmer_count	2
	max_kmer_count	200
GraphGen	min_shared_kmers	2
	max_degree	25
GraphLPA	min_shared_kmers	2
	max_shared_kmers	20000
	$min\_reads\_per\_cluster$	2
GlobalCluster	$size$	100
	k_mer_count	100
	$cosine$	0.85～1

3 超参数算法实现

3.1 算法实现流程

Hyperopt 是一个用于超参数优化的开源软件[12]，其使用 TPE 作为代理函数，用于优化聚类结果评估函数[13]。Hyperopt 主要包括 fn，$space$，$algo$，max_evals 4 个参数。其中，fn 定义 SpaRC 算法实现函数，$space$ 定义超参数集搜索空间，$algo$ 定义代理函数模型(Tree Parzen Estimator)，max_evals 定义 SpaRC 算法最大迭代次数。基于 TPE 的 SpaRC 超参数优化模型实现过程如下：

1)建立目标函数替代概率模型，找到使代理函数表现最佳的超参数。使用 TPE 建模策略建立目标函数概率替代模型，用该替代概率模型在搜索空间内选择表现最佳的超参数。

2)将选择的超参数应用于 SpaRC 算法，根据已知答案评价聚类结果。将模型搜索到的最佳超参数应用于 SpaRC 算法实现，并对算法实现结果进行评价。

3)根据步骤 2)中的评价结果更新包含新结果的代理模型。通过 TPE 建模策略转换生成过程来模拟 $p(x|y)$，用非参数密度替换原先配置分布来完成代理模型更新。最后进行如下替换：均匀→截断高斯混合，对数均匀→指数截断高斯混合，分类→再加权分类。其中模型的更新替代通过式(1)的密度函数来完成。

$$p(x|y)=\begin{cases} l(x), & \text{if } y<y^* \\ g(x), & \text{if } y\geqslant y^* \end{cases} \tag{1}$$

其中，$l(x)$是通过$\{x^{(k)}\}$得到的密度；$g(x)$是通过剩余观察值得到的密度。

4)重复步骤 2)和步骤 3)，直至达到最大迭代次数。

通过上述过程能够找到使聚类结果评估函数最优的超参数集。

3.2 评估指标

实验数据集是来自 PacBio 测序平台的两组长序列片段(long reads)[14]和来自 Critical Assessment of Metagenome Interpretation(CAMI2)平台的一组短序列片段(short reads)[15]，这 3 组数据集均是已经标记过的数据集，可以用外部标准来评估 SpaRC 聚类性能。评价指标有 ARI、NMI、FMI、F1-Score、纯度、完整性等[16]，评价指标的选择会对实验结果产生一定的影响。通过实验验证，最终选择纯度和完整度作为目标函数的评价指标。纯度是指能够将每个测序片段正确分配到相应簇之中的比例；完整度是指标记过的基因组(genome)中被分到正确测序片段的比例。纯度和完整度的计算公式如式(2)所示：

$$(purity|comple)(\Omega,\mathbb{C})=\frac{1}{N}\sum_k \max_j |\omega_k \cap c_j| \tag{2}$$

其中，$\Omega=\{\omega_1,\omega_2,\cdots,\omega_k\}$是簇集合；$\mathbb{C}=\{c_1,c_2,\cdots,c_j\}$是标记基因组(genome)集合。

根据 SpaRC 算法的参数特性，LocalCustering 采用纯度中位数、完整度中位数、纯度百分比以及完整度百分比四者的平均值作为最终评价指标；GlobalClustering 采用纯度中位数、完整度中位数二者的平均值作为最终评价指标。

4 实验结果与分析

4.1 实验环境

参数优化实验是在 Amazon Web Service(AWS)的 ElasticMapReduce(EMR，emr-5.17.0)上执行的。根据数据集的大小，使用不同数量的 r4.2xlarge 实例来形成一个集群，详细的配置信息如表 4 所列。

表 4 AWS EMR 的配置

Table 4 Configuration of AWS EMR

参数	设置
# of cores/node	8
Memory/node	61
Storage/node	160 GB SSD
Ethernet	10 Gbps
Spark version	2.3.1
Hdoop version	2.8.3
Cluster mode	YARN
# of executors/node	3
Driver memory/GB	40
Driver cores	5
Memory/executor/GB	16
Cores/executor	2
HDFS Block Size/MB	32

4.2 数据集

为了测试参数的优化效果，选取了两组长序列片段(ActinoMock 和 transcriptome)和一组短序列片段(CAMI2)来进行实验。实验数据集如表 5 所列。

表 5　数据集

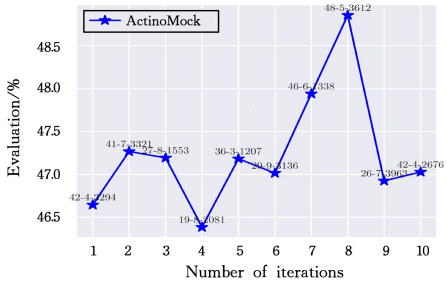
Table 5　Datasets

数据集	测序片段长度/bp	测序片段数量	数据集大小/GB
ActinoMock	50～45 165	389 806	2.4
transcriptome	300～30 816	1 107 889	3.8
CAMI2	150＊2	83 029 581	25

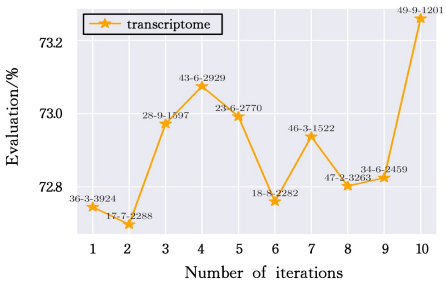
4.3　实验结果

LocalClustering 包括 KmerMapReads，GraphGen，GraphLPA，CCAddSeq 4 个模块。为了节约计算资源和时间，分别对两个数据集进行采样。其中，ActinoMock 数据集采样后约为 38 000 条序列片段(reads)，transcriptome 数据集采样后约为 59 000 条序列片段(reads)，采样后的测序片段数约占总测序片段数的 10%。然后通过超参数优化模型对采样后的数据进行超参数选择，随着迭代次数增多，得到的超参数会越来越接近最优的超参数。图 1 和图 2 为超参数优化模型搜索过程中目标函数的变化趋势。

图 1 中每个点的标注格式为 *k*-*min_shared_kmers*-*max_kmer_count*。为了防止资源浪费，*max_evals* 参数设置为 10，即允许最大迭代次数为 10。从图 1(a)可以看出，ActinoMock 数据集在第 8 次迭代时得到最佳超参数 *k* 为 48，*min_shared_kmers* 为 5，*max_kmer_count* 为 3 612，目标函数的值为 48.855%；从图 1(b)可以看出，transcriptome 数据集在第 10 次迭代时得到最佳超参数 *k* 为 49，*min_shared_kmers* 为 9，*max_kmer_count* 为 1 201，目标函数的值为 73.26%。然后根据得到的超参数对 SpaRC 算法进行参数设置，并对原始的数据集进行聚类分析。



(a) ActinoMock



(b) Transcriptome

图 1　LocalClustering 目标函数的变化趋势

Fig. 1　Change trend of LocalClustering objective function

表 6、表 7 列出了优化前后原始数据集的评价指标对比。可以看出，ActinoMock 数据集评价指标由优化前的 44.1575%提高到 46.9950%，transcriptome 数据集评价指标由优化前的 69.5225%提高到 71.025%。

表 6　ActinoMock 数据集

Table 6　ActinoMock dataset

序号	*k*	*min_shared_kmers*	*max_kmer_count*	评价指标/%
优化前	19	8	2 081	44.157 5
优化后	48	5	3 612	46.995 0

表 7　transcriptome 数据集

Table 7　transcriptome dataset

序号	*k*	*min_shared_kmers*	*max_kmer_count*	评价指标/%
优化前	17	7	2 288	69.522 5
优化后	49	9	1 201	71.025

表 8 列出了优化前后 ActinoMock 数据集和 transcriptome 数据集各项指标的对比。*Med_P* 代表纯度中位数，100%_*P* 代表纯度百分比，*Med_C* 代表完整度中位数，100%_*C* 代表完整度百分比。可以看出，ActinoMock 数据集优化前后纯度百分比和完整度中位数变化较大，而纯度中位数和完整度百分比变化较小；transcriptome 数据集优化前后纯度中位数和纯度百分比变化较大，而完整度中位数和完整度百分比变化较小。transcriptome 数据集物种丰富度较高，通过参数设置不容易影响其完整度变化，因此 transcriptome 数据集优化前后完整度中位数和完整度百分比变化较小。通过实验可以发现，超参数优化模型能够在 SpaRC 算法对短序列片段聚类过程中选择出最佳的超参数集，对 SpaRC 算法起到良好的优化效果。

表 8　LocalClustering 优化前后各项指标的对比

Table 8　Comparison of various indicators before and after LocalClustering optimization

数据集	序号	*Med_P*	100%_*P*	*Med_C*	100%_*C*
ActinoMock	前	100.00	74.92	1.71	0.00
	后	100.00	79.26	8.72	0.00
transcriptome	前	68.66	16.78	100.00	92.65
	后	71.43	19.88	100.00	92.79

GlobalClustering 包括 MinimizerMapReads，GraphGen，GraphLPA，GlobalCluster，CCAddSeq 5 个模块，其中 MinimizerMapReads，GraphGen，GraphLPA，CCAddSeq 4 个模块与 LocalClustering 相似。通过实验发现，LocalClustering 参数设置对短序列片段(short reads)段并不敏感，因此在 GlobalClustering 中涉及到 LocalClustering 的参数均按 SpaRC 官方文档提供的默认参数设置。

图 2　GlobalClustering 目标函数的变化趋势

Fig. 2　Change trend of GlobalClustering objective function

图 2 中每个点的标注格式为 *cosine*。为了防止资源浪费，*max_evals* 参数同样设置为 10，即允许最大迭代次数为 10。图 2 给出了超参数优化模型在搜索过程中目标函数的变

化趋势。可以看出，CAMI2 数据集在第 2 次迭代时得到最佳超参数 *cosine* 为 0.907，目标函数的值为 60.135%。

表 9 列出了优化前后 CAMI2 数据集各项指标的对比。可以看出，CAMI2 数据集优化前后纯度中位数和纯度百分比变化较大，而完整度中位数和完整度百分比变化较小。

表 9 GlobalClustering 优化前后各项指标的对比

Table 9 Comparison of various indicators before and after GlobalClustering optimization

数据集	序号	*Med_P*	100%_*P*	*Med_C*	100%_*C*
CAMI2	前	30.51	7.30	22.12	0.00
	后	98.90	31.25	21.37	0.13

图 3 给出了 CAMI2 数据集优化前后纯度在 90%以上，完整度在 20%～90%区间内基因组(genome)的数量变化趋势。可以看出，优化前后完整度达到 50%，70%，90%的基因组(genome)数量没有变化，但是优化后完整度达到 20%的基因组(genome)数量多于优化前。

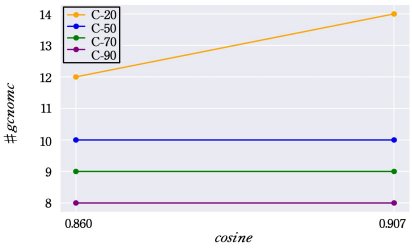
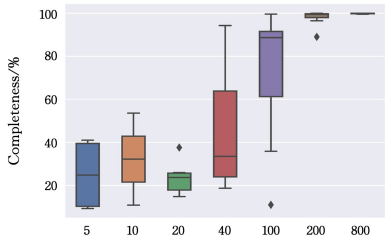


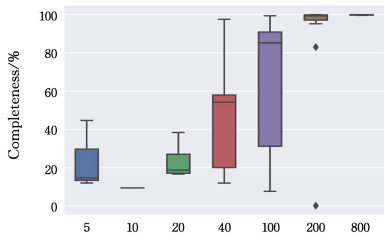
图 3 genome 的数量变化趋势

Fig. 3 Change trend of number of genomes

图 4 给出了优化前后纯度在 90%以上的基因组(genome)覆盖度与完整度的分布情况。生物学上一般认为覆盖度在 10x 以上，基因组测序错误率才能够得以保证，因此仅考虑覆盖度在 10x 以上的基因组。从图 4 可以看出，覆盖度在 20x 以下和 40x～100x 区间内时，优化前后完整度的中位数变化不大；覆盖度在 20x～40x 区间内时，优化前完整度的中位数在 40%以下，优化后完整度中位数在 50%以上，优化后完整度明显优于优化前。



(a)Coverage-0.860



(b)Coverage-0.907

图 4 优化前后覆盖度与完整度分布

Fig. 4 Coverage and completeness distribution before and after optimization

图 5 给出了为优化前后纯度和完整度的整体分布情况。可以看到，优化前后完整度整体分布变化不明显；优化前纯度整体分布在 25%左右，优化后纯度整体分布在 75%以上。因此，优化前后在完整度整体分布基本不变的情况下，纯度整体分布有了较大提升。

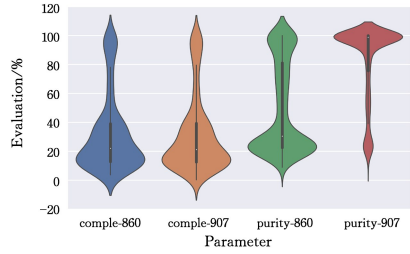


图 5 优化前后纯度和完整度整体分布

Fig. 5 Overall distribution of purity and integrity before and after optimization

图 6 给出了优化前后各个数据集的指标变化情况。从图 6 可以更加直观地看到，基于 TPE 策略的 SpaRC 超参数优化模型对于 SpaRC 算法参数起到良好的优化效果，GlobalClustering 优化效果明显优于 LocalClustering。其中，LocalClustering 主要对长序列片段(long reads)进行聚类，长序列片段产生 k-mer 数量较多，通过共享 k-mer 数能够更加容易地判断两条序列片段之间的相似度；GlobalClustering 主要对短序列片段(short reads)进行聚类，短序列片段产生 k-mer 数量较少，通过共享 k-mer 数不容易判断两条序列片段之间的相似度，因此，GlobalClustering 对参数设置更加敏感。

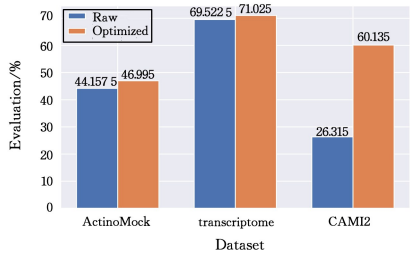


图 6 优化前后各个数据集的指标变化

Fig. 6 Changes in indicators of each data set before and after optimization

综上，可以得出结论：Hyperopt 超参数优化模型在 SpaRC 算法上能够起到良好的优化效果，在保证算法效果的同时，能够有效地节约时间和计算资源。

4.4 模型对比

为了比较不同的超参数优化模型在 SpaRC 算法上的性能差别，选择 Speartmint 参数优化模型进行对比实验。Spearimint 是一个基于高斯过程回归的标准贝叶斯超参数算法优化库，应用广泛，在低维空间表现出色。

首先，分别通过 Hyperopt 模型(运行时间为 100 min)和 Spearmint 模型(运行时间为 120 min)得到两组参数，然后将两组参数分别用于表 5 的数据集进行实验。图 7 给出了两组实验各种指标的对比，其中，*M_Purity* 代表纯度中位数，100_*Purity* 代表纯度百分比，*M_comple* 代表完整度中位数，100_*comple* 代表完整度百分比。可以看出，Hyperopt 模型和

Spearmint 模型在实验数据集上的效果差别不大，但是 Hyperopt 模型的运行时间要比 Spearmint 模型短。

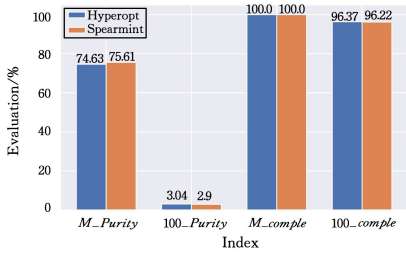


图 7　模型指标对比

Fig. 7　Model index comparison

结束语　本文针对 SpaRC 算法的参数选择问题，通过 TPE 超参数优化（先对数据集采样，然后在采样数据集上进行参数选择，最后将选择的参数应用在原始数据集上），使得 SpaRC 算法在消耗较少计算资源和时间的情况下得到相对较好的超参数，从而避免了资源浪费；同时在调整参数过程中 TPE 超参数优化模型还可以根据不同的数据集自行设置参数探索次数。实验结果证明，本文提出的优化方法在改善 SpaRC 算法性能方面有着良好的效果。

Spark 是专为大规模数据处理而设计的快速通用计算引擎，基于 Spark 的应用程序参数设置对于计算资源的有效利用至关重要。如果能够将机器学习算法和超参数优化相结合来有效地选择合适的参数，就可以在保证结果的基础上有效节约计算资源和节省时间。在接下来的工作中，可以探索更多关于超参数优化的算法，以找到最适合 SpaRC 的超参数优化算法。

参 考 文 献

[1] MARTIN H, MANJA M. De novo transcriptome assembly: A comprehensive cross-species comparison of short-read RNA-Seq assemblers[J]. Giga Science, 2019, 8(5): 39.

[2] QUINCE C, WALKER A, SIMPSON J, et al. Shotgun metagenomics, from sampling to analysis[J]. Nat Biotechnol, 2017, 35: 833-844.

[3] LENCZ T, YU J, PALMER C, et al. High-depth whole genome sequencing of an Ashkenazi Jewish reference panel: enhancing sensitivity, accuracy, and imputation[J]. Human Genetics, 2018, 137(4): 343-355.

[4] BERTRAND D, SHAW J, KALATHIYAPPAN M, et al. Hybrid metagenomic assembly enables high-resolution analysis of resistance determinants and mobile elements in human microbiomes[J]. Nature Biotechnology, 2019, 37(8): 937-944.

[5] LI D H, LIU C M, LUO R B, et al. MEGAHIT: An ultra-fast single-node solution for large and complex metagenomics assembly via succinct de bruijn graph [J]. Bioinformatics, 2015, 31(10): 1674-1676.

[6] GUO X, YU N, DING X J, et al. DIME: A novel framework for de novo metagenomic sequence assembly[J]. Jouranl of Computational Biology, 2015, 22(2): 159-177.

[7] SHI L Z, MENG X D, TSENG E, et al. SpaRC: Scalable Sequence Clustering using Apache Spark[J]. Bioinformatics, 2019, 35(5): 760-768.

[8] SUN Y, XUE B, ZHANG M, et al. An experimental study on hyper-parameter optimization for stacked auto-encoders[C]// 2018 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2018: 1-8.

[9] ZHOU Z H. Machine Learning[M]. Beijing: Tsinghua University Press, 2016: 147-162.

[10] GHANBARI-ADIVI F, MOSLEH M. Text Emotion Detection in Social Networks Using a Novel Ensemble Classifier Based on Parzen Tree Estimator (TPE)[J]. Neural Computing and Applications, 2019, 31(12): 8971-8983.

[11] RAGHAVAN U N, ALBERT R, KUMARA S. Near linear time algorithm to detect community structures in large-scale networks[J]. Physical Review Research, 2007, 76(3): 036106.

[12] BERGSTRA J, KOMER B, ELIASMITH C, et al. Hyperopt: A Python library for model selection and hyperparameter optimization[J]. Computational Science & Discovery, 2015, 8(1): 014008.

[13] BERGSTRA J, BARDENET R, BENGIO Y, et al. Algorithms for Hyper-Parameter Optimization[J]. Advances in Neural Information Processing Systems, 2011, 24: 2546-2554.

[14] YANG L, SHAMI A. On hyperparameter optimization of machine learning algorithms: Theory and practice[J]. Neurocomputing, 2020, 415: 295-316.

[15] SCZYRBA A, HOFMANN P, BELMANN P, et al. Critical Assessment of Metagenome Interpretation-a benchmark of metagenomics software[J]. Nature Methods, 2017, 14: 1063-1071.

[16] BHASKAR M, NICK C. An Introduction to Neural Information Retrieval[M]. America: Now Publishers, 2018: 11-19.

DENG Li, born in 1978, associate professor. Her main research interests include metagene data analysis and machine learning.