基于自注意力机制的牛津纳米孔碱基测序信号识别

谢少辉

摘要

随着测序技术的发展，越来越多的基因组分析项目开始采用三代测序。牛津纳米孔测序技术是三代测序中最具有代表性的，可以实现低成本的长读实时测序。然而，纳米孔测序需要从嘈杂的电信号中识别碱基，识别过程带来的高错误率是限制纳米孔测序进一步应用的主要因素。在本文中，我们复现一个基于自注意力机制的端到端碱基识别神经网络模型 SACall，对信号的全局上下文相关性进行建模，最终通过束搜索算法从概率中解码得到碱基序列。我们还提出一个改进版本 SACall-Conv，通过融合全局和局部上下文信息来提高模型的表达力。我们在九个细菌样本组成的数据集上，对不同的碱基识别模型（包括 SACall，SACall-Conv，Guppy 和 Guppy-KP）进行了基准测试，从识别准确率和组装质量对模型性能进行评估。结果表明，SACall 相比其他非自注意力机制的模型具有更好的识别准确率和更高的组装一致性，改进后的 SACall-Conv 则进一步提高了模型的精度。

关键词：纳米孔测序；自注意力机制；碱基识别

1 引言

基因组测序是生物医学和生物信息学中最基础的一项技术，所有跟基因组相关的研究都离不开测序技术的进步与发展。从基于荧光标记的 Sanger 测序，到基于循环阵列合成的第二代高通量测序技术[1]，再到以纳米孔测序技术为代表的第三代测序技术[2]，测序读长由短变长，通量由低到高。相比高通量测序技术，纳米孔测序技术读长更长（最高可达到 10kbp），有助于解决 DNA 长重复片段检测[3]和甲基化修饰[4]等问题。

纳米孔测序时，通过马达蛋白牵引核酸序列进入纳米孔蛋白并且测量出每个 kmer（连续 k 个碱基构成的子串）的过孔信号[5]，然后将信号翻译为碱基序列。由于碱基序列过孔时速率不稳定，加上环境影响和仪器自身的噪声，给分析信号带来了较大的挑战。早期的碱基信号识别软件通过分割信号以及隐马尔可夫模型来实现，这种方案准确率低且容易造成信息丢失。随着深度学习的发展，越来越多的算法采用深度神经网络直接处理原始信号，实现了端到端的碱基信号识别，同时提升了分析的效率和准确率。

本次课程的论文复现工作拟通过对纳米孔原始信号进行处理，采用基于自注意力机制的深度神经网络得到测序结果，再通过基准测试数据集对结果的各项指标进行充分验证。

2 相关工作

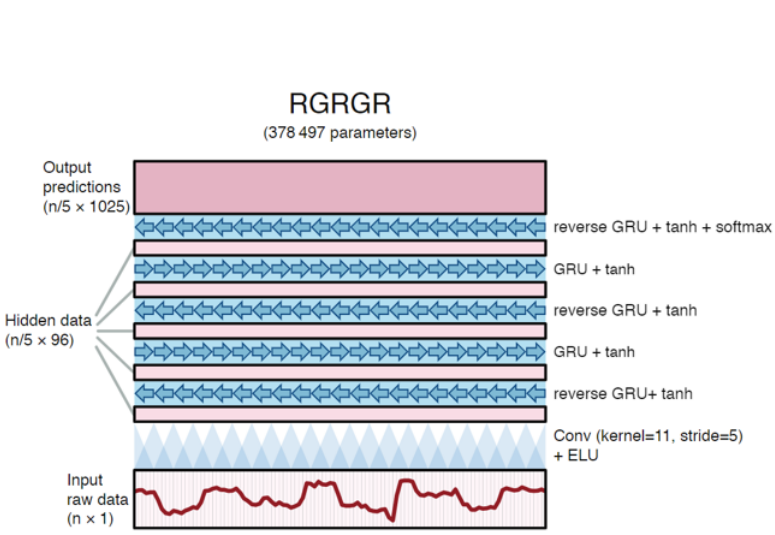
2.1 基于事件分割的识别模型

目前纳米孔测序仪的电信号频率普遍为 4kHz，而单链 DNA 分子以每秒 450 个碱基的平均速度通过纳米孔，意味着每个碱基的观察信号点大概为 9 个，这一段相对平稳的信号称为一个事件（event）。早期的纳米孔测序信号识别软件都是基于事件分割的方法，包括 Metrichor、Nanocall[6]和 DeepNano[7]。Metrichor 是牛津纳米孔公司开发的第一个云平台测序信号识别软件，自 2017 年已经停

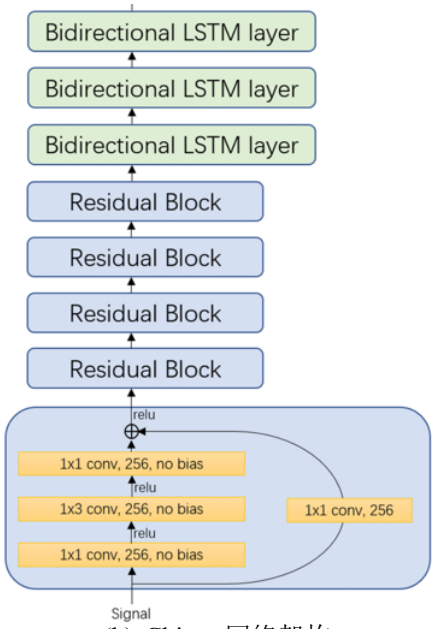
止使用。Nanocall 和 DeepNano 都是第三方研究机构开发的开源软件。Nanocall 使用隐马尔科夫模型（HMM）来预测碱基序列，模型的隐藏状态表示所有可能的事件，然后在训练数据上训练事件之间的转移概率。DeepNano 则将每个事件的信号均值/方差作为输入，通过双向循环神经网络[8]预测每个事件对应的碱基概率。

2.2 基于神经网络的端到端识别模型

基于事件分割的碱基信号识别模型，其性能完全依赖于事件分割的质量，在分割过程中也会造成信息的缺失并引入错误，导致识别准确率下降。随着深度神经网络逐渐成熟，端到端的测序信号识别模型已经成为主流，纳米孔公司也提供了一系列基于神经网络的测序识别软件，包括 Albacore，Scrappie，Guppy 等。其中，Guppy 是目前应用最广泛的商业闭源软件，采用 RGRGR 模型，即五层方向交替变换的门控循环单元 GRU[9]堆叠形成的网络结构，如图 1(a)所示。Chiron[10]是第三方研究机构开发的端到端识别模型，由多层卷积残差网络[11]和双向 LSTM 网络[12]构成，如图 1(b)所示。然而，Chiron 的运行速度非常慢，实用性较差，只能处理较小规模的测序数据集[13]。



(a) Guppy 网络架构



(b) Chiron 网络架构

图 1: 基于神经网络的端到端识别模型

3 本文方法

3.1 本文方法概述

本文提出了 SACall，一种基于自注意力机制的端到端碱基测序信号识别模型，可以直接从原始电信号中识别碱基序列，而无需进行信号分割。SACall 的网络结构如图 2所示，由卷积模块、全局上下文信息提取模块（六层 Tranformer 编码器[14]）以及最终的解码模块（线性分类器）构成，使用连接主义时间分类[15]（CTC）损失作为模型的优化目标。

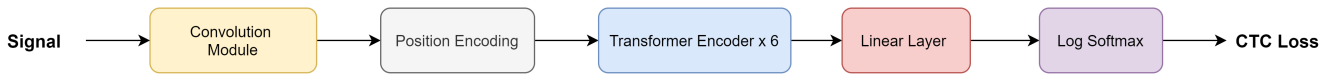


图 2: SACall 网络架构

3.2 卷积模块

卷积模块由两个卷积层构成，用于对输入信号进行降采样并且提升信号的特征维度，如图 3所示。其中第一个卷积层将信号长度减半，将输入信号特征维度提升到 128；第二个卷积层将信号长度变成原来的四分之一，将输入信号特征维度提升到 256。每个卷积层由一维卷积 Conv、批归一化 BatchNorm[16]和线性整流函数 ReLU[17]构成。一维卷积的计算过程与常用的二维卷积基本一致；批归一化处理可以保证不同层训练参数分布的一致性，使训练更加稳定；线性整流函数避免了部分梯度爆炸和梯度消失的问题。

图 3: 卷积模块

3.3 全局上下文特征提取模块

卷积模块后是位置编码和多层 Transformer 编码器堆叠而成的全局特征提取模块。每一层 Transformer 编码器的网络结构如图 4所示，由一个带残差结构的多头自注意力模块和前馈网络模块构成。

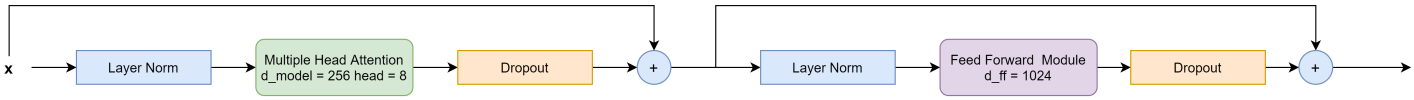


图 4: 全局上下文特征提取模块

3.3.1 位置编码

在使用多头自注意力模块提取信号的全局上下文特征前，需要先进行位置编码，即给信号的每个位置提供一个独一无二的位置向量，然后融合到卷积层的输出向量中。这是因为多头自注意力模块是完全并行的，无法捕获原始信号的位置信息，需要人为进行干预。基于原始 Transformer 编码器的设计，位置编码向量可以用一组交替的正余弦函数族计算，公式如下：

$$PositionEncoding_t[2i] = sin(\frac{t}{10000^{\frac{2i}{d_{model}}}})$$

$$PositionEncoding_t[2i+1] = cos(\frac{t}{10000^{\frac{2i}{d_{model}}}})$$

$$Embedding_t = Embedding_t + PositionEncoding_t$$

其中 $PositionEncoding_t$ 代表某个时间戳的位置向量，$PositionEncoding_t[2i+1]$ 和 $PositionEncoding_t[2i]$ 代表位置向量在奇偶索引的值，d_{model} 是位置向量的特征维度。因为位置向量和卷积层的输出向量维度相同，我们可以将这两个向量直接相加，得到带有位置信息的特征向量。

3.3.2 多头自注意力模块

得到位置编码的特征向量后，我们用多头自注意力模块来计算序列中任意两个信号特征之间的相似性，以此表示信号的全局上下文相关信息。在本文中，我们使用带缩放的点乘自注意力策略（Scaled Dot-Product Attention），如图 5(a)所示。对于输入的特征向量序列 X，我们用三个不同的线性变换将 X 分别变换为查询 Q、键 K 和值 V，然后通过点乘操作计算两两位置的查询向量和键向量的相似度，以此作为权重对不同位置的值向量进行加权求和，计算过程如下：

$$Q = W^Q X$$

$$K = W^K X$$

$$V = W^V X$$

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_{model}}})V$$

其中，$W^Q \in R^{d_{model} \times d_{model}}$，$W^K \in R^{d_{model} \times d_{model}}$，$W^V \in R^{d_{model} \times d_{model}}$

多头自注意力是在原始自注意力机制的基础上，计算多个子空间变换下的全局特征表示，然后将这些特征合并作为最终的输出，如图 5(b)所示。多头自注意力机制允许模型捕获不同层次的上下文信息，计算过程如下：

$$MultiHead(Q, K, V) = Concat(head_1, head_2, \ldots, head_h)W^O$$

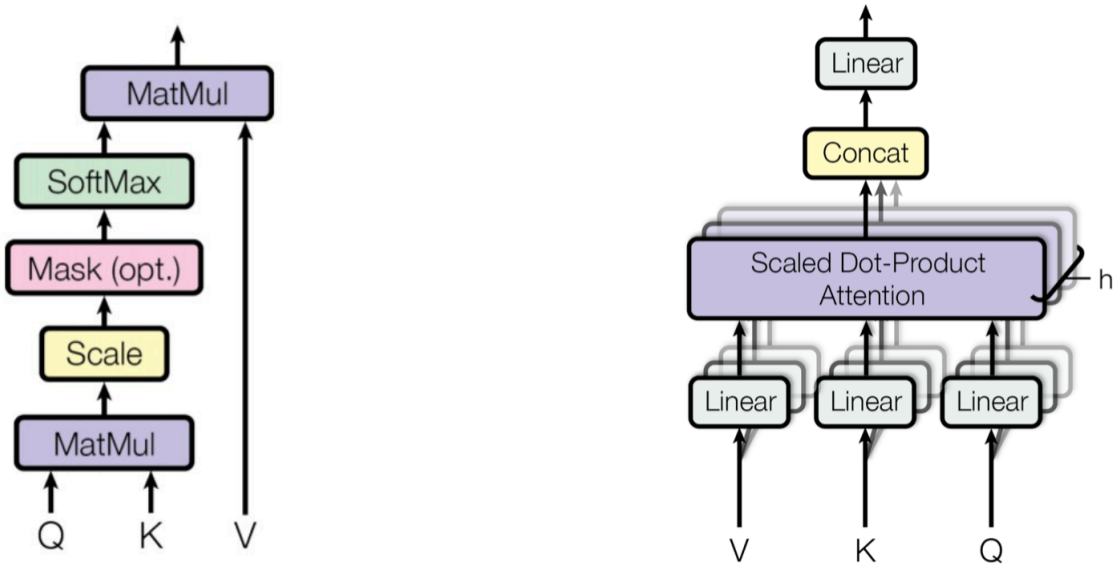
$$Q_i = W_i^Q X$$

$$K_i = W_i^K X$$

$$V_i = W_i^V X$$

$$head_i = Attention(Q_i, K_i, V_i)$$

其中，$W_i^Q \in R^{\frac{d_{model}}{h} \times d_{model}}$，$W_i^K \in R^{\frac{d_{model}}{h} \times d_{model}}$, $W_i^V \in R^{\frac{d_{model}}{h} \times d_{model}}$, $W^O \in R^{d_{model} \times d_{model}}$，$h$ 是头的数量。



(a) 自注意力机制　　(b) 多头自注意力机制

图 5: 自注意力机制示意图

3.3.3 前馈网络模块

Transformer 编码器的最后是前馈网络模块，如图 6所示，由两个带 ReLU 激活函数的线性层构成，这是为了在模型中加入非线性变换以提高模型的表达力。前馈网络的第一个线性层将输入的全局特征向量进行特征融合，并将其特征维度从 d_{model} 变成 d_{ff}，第二个线性层则将特征维度从 d_{ff} 还原回 d_{model}，这样就能作为下一个 Transformer 编码器的输入。

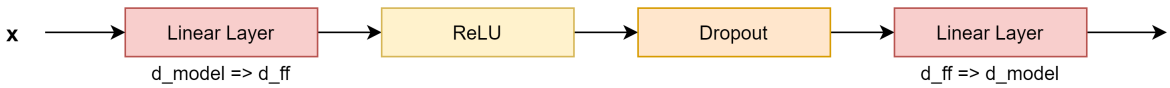


图 6: 前馈网络

3.4 解码模块

SACall 网络架构的最后一层是解码模块，将前面得到的与电信号相关的高维特征表示 X，通过一个线性分类器和 softmax 函数转为对应位置上碱基发生的概率值，解码模块的计算如下：

$$P(o_t = c|X) = \frac{e^{W_c h_t}}{\sum_c e^{W_c h_t}}$$

其中 o_t 代表在时间戳 t 预测的碱基，h_i 代表在时间戳 t 的高维特征表示，W_c 是解码模块的线性分类器参数，碱基 c 的字符集为 $(A, T, C, G, -)$，$-$ 代表空字符。

3.5 CTC 损失函数的定义

大部分情况下，即使是经过降采样的电信号，其长度依然大于这个电信号对应的碱基序列，因此我们需要将网络输出的碱基概率值跟真实的碱基序列进行对齐，才能计算最终的损失，这个函数可以用连接主义时间分类函数（CTC）来定义。对于一个输入的信号特征序列 X，其预测的碱基序列为 O（长度与 X 一致），真实的标签序列为 Y（长度通常小于 O），CTC 损失函数为：

$$loss = -log(P(Y|X))$$

$$P(Y|X) = \sum_{O \in \Omega} P(O|X) = \sum_{O \in \Omega} \prod_{t=1}^{T} P(o_t|X)$$

其中，Ω 代表所有长度为 $|O|$ 的序列中，可以跟 Y 对齐的序列（一般称为 CTC 字符串）集合。这里的对齐操作包括先合并序列中的重复字符，再删除序列中的空字符，例如 a - t t a - - a 合并重复字符后得到 a - t a - a，再删除序列中空符号得到 ataa，从而视为与序列 ataa 对齐的一个 CTC 串。

3.6 束搜索解码

训练得到的模型在推理时，需要利用解码模块得到的碱基概率，推断出最优的碱基序列。因为解码搜索空间随时间序列长度呈指数型增长，不存在可行的全局最优算法，因此本文中我们采用启发式的束搜索解码算法，按时间顺序迭代地选择每个时间戳最优的几个候选序列（候选序列的个数为束搜索的大小），最后选择得分最高（条件概率最大）的序列作为我们预测的碱基序列。图 7是束搜索的一个例子（束搜索大小为 2），T=1 时，我们挑选得分最高的两个碱基 A 和 G（用红色标注），并以此为基础扩展搜索空间；T=2 时，候选序列变成 A-和 GA（用红色标注）；T=3 时，候选序列变成 A-C 和 A-T（用红色标注）。

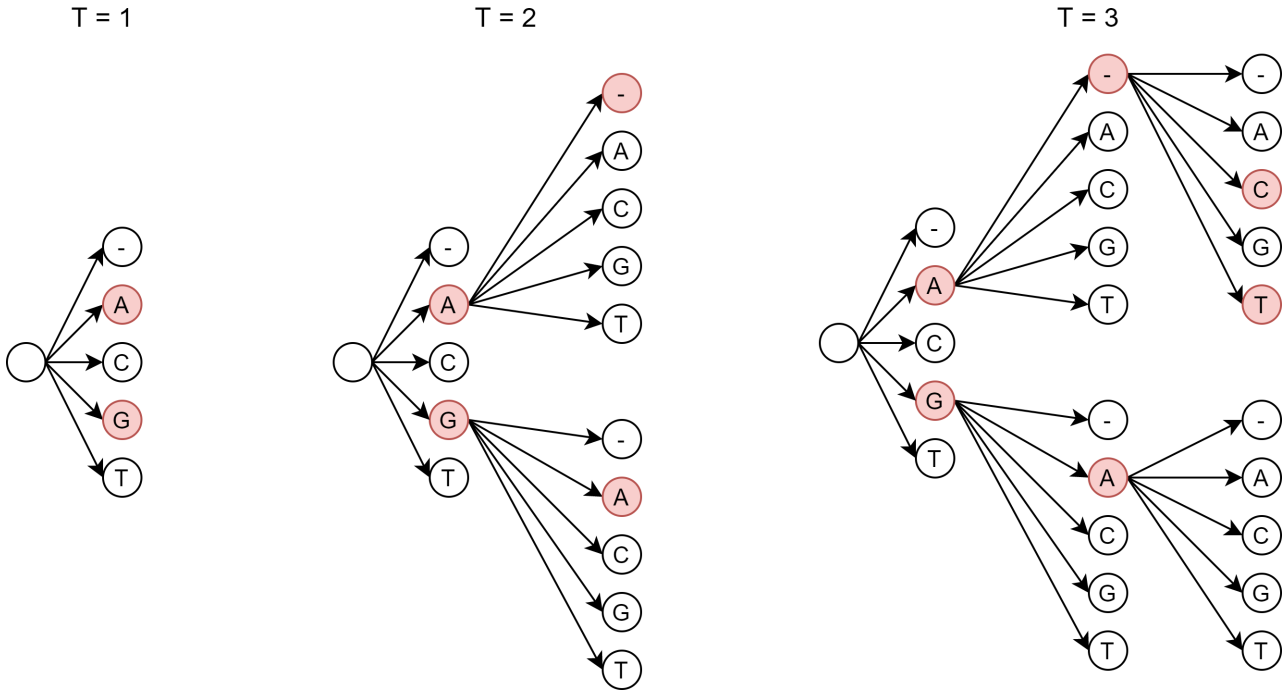


图 7: 束搜索示例

4 复现细节

4.1 与已有开源代码对比

本文在复现时，对测序文件（Fast5 文件）的预处理采用了纳米孔公司开源的 ont-bonito（https://github.com/nanoporetech/bonito）的部分代码，参考了 SACall（https://github.com/huangnengCSU/SACall-basecaller）的部分模型源代码。在 SACall 的基础上，我们提出一个改进版本 SACall-Conv。与 SACall 不同的是，SACall-Conv 在原来的 Transformer 编码器中引入了卷积模块，通过融合全局上下文和局部上下文特征的方式，进一步提升模型的表达能力。图 8显示了 SACall 和 SACall-Conv 中

Transformer 编码器的区别。

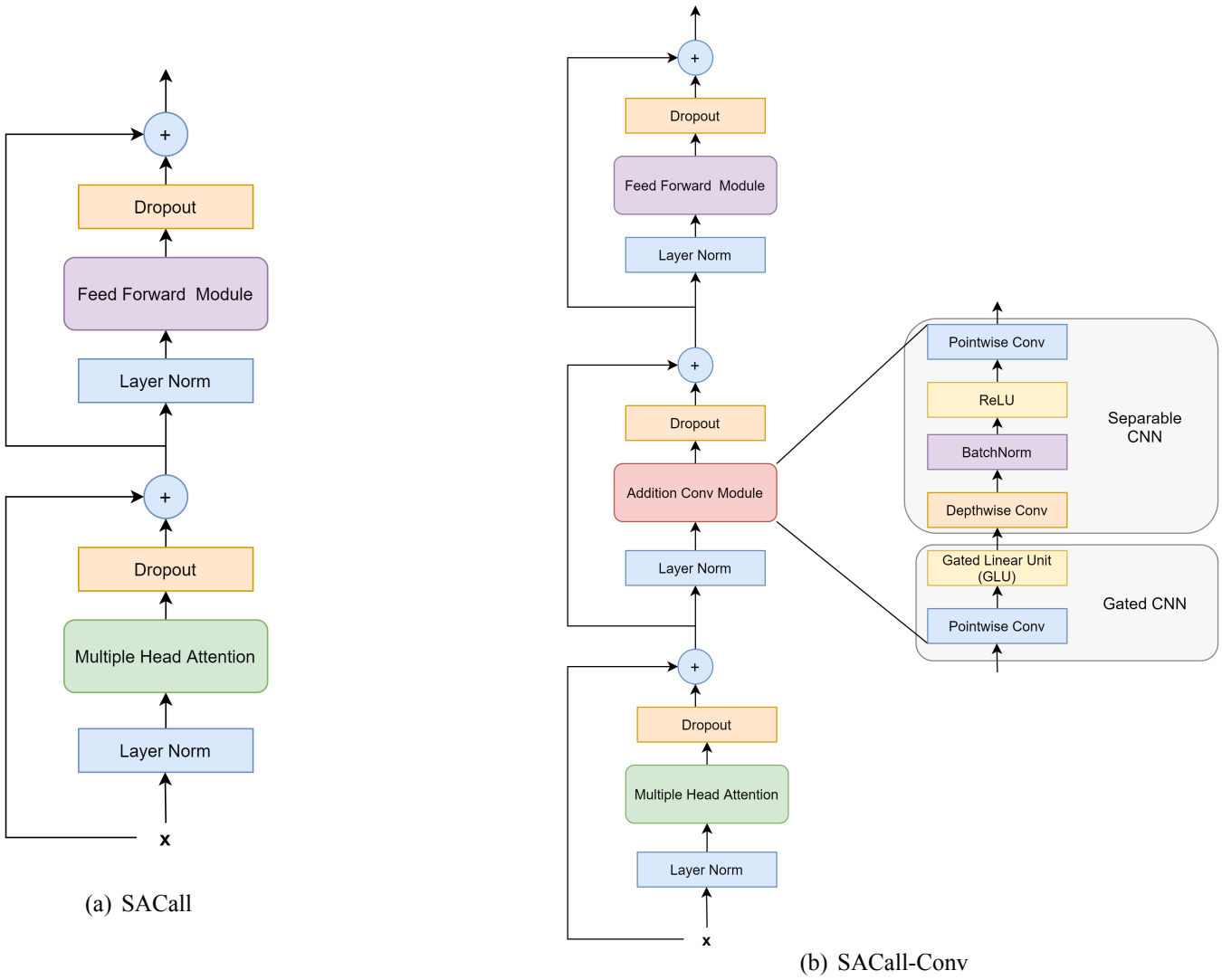


图 8: SACall 和 SACall-Conv 的 Transformer 编码器区别

SACall-Conv 在多头自注意力模块之后加入了一个卷积模块，用于提取信号之间的局部上下文特征。额外增加的卷积模块由两个部分构成，分别是带门控的卷积模块和可分离卷积模块。门控卷积模块由一个逐点卷积和一个门控线性单元组成，用于融合并选择全局特征中不同维度的信息，类似于前馈网络模块的第一个线性层。可分离卷积模块主要由串联的深度可分离卷积和逐点卷积构成。相比于传统的卷积操作，可分离卷积能够减少更多的计算量，训练和推理效率更高，而且在精度上也不会有太多的损失。

4.2 实验环境搭建

本文的训练和测试均在 Linux 操作系统执行，采用 Pytorch 框架实现模型代码，使用 anaconda3 包管理器部署环境。训练数据集采用纳米孔公司的 ont-bonito 软件包提供的 dna R9.4.1 数据集（包含 1221470 条长度为 3600 的信号数据）。所有模型均采用 AdamW 优化器（参数 $\beta_1 = 0.9, \beta_2 = 0.999$，$\varepsilon = 1e-8$，weight_decay=0.01，$lr = 1e^{-4}$），批大小为 64，迭代循环训练 30 次。

4.3 纳米孔碱基测序信号识别流程

图 9展示了从纳米孔原始测序文件（Fast5）得到碱基序列的完整流程。首先需要将数字电信号数据转为电流值（单位 pA），转换的公式如下（其中 offset、range、digitisation 都是 Fast5 文件中的参数）：

$$signal = (data + offset) * range/digitisation$$

接着需要对电信号进行标准化处理，我们使用中位数标准差（MAD）进行标准化，可以避免异常值的影响，标准化公式如下：

$$signal = (signal - median(signal))/MAD(signal)$$

最后将信号按照固定长度进行分段（chunknize），对每个片段独立并行地进行识别，最后合并所有片段的结果得到信号对应的碱基序列。

图 9: 纳米孔测序信号识别流程

4.4 创新点

本文第一次提出采用基于自注意力机制的深度学习模型进行纳米孔碱基信号识别，通过自注意力层来捕获不同位置信号的相关性，从而提供了具有全局上下文的表征能力。

相比于其他基于循环神经网络、卷积神经网络的模型，本文提出的方案能提供更高的识别准确率，同时也提升了下游基因组组装的质量，有应用于其他场景（例如碱基修饰检测）的潜力。

5 实验结果分析

我们采用论文[13]提供的九个细菌样本作为基准测试集，对 Guppy2.2.3（纳米孔官方提供的闭源软件）、Guppy-KP（论文[13]中采用 Guppy 网络结构重新训练的模型）、SACall 和 SACall-Conv 进行了性能比较。基准测试数据集的基本信息如表格 1所示。

基准测试的评价指标包括识别准确率以及碱基序列组装的质量，这需要通过一些额外的分析过程得到。基准测试的分析流程如图 10所示。首先，我们需要从原始的纳米孔测序文件（Fast5）中提取电信号并识别得到碱基序列，然后通过 minimap2 工具[18]将碱基序列与对应的参考基因组进行比对，通过比对的结果来计算碱基识别的准确率。另外，我们使用 Flye[19]工具得到碱基序列的组装重叠群（contig），借助 Quast[20]评估组装的质量，同时我们也将重叠群与参考基因组进行比对，从而得到组装的准确率。初步组装得到的重叠群一般带有比较多的错误，我们进一步使用 medaka（https://github.com/nanoporetech/medaka）工具对重叠群进行纠错，然后随机采样其中一部分片段，通过 minimap2 比对得到纠错后的组装准确率。

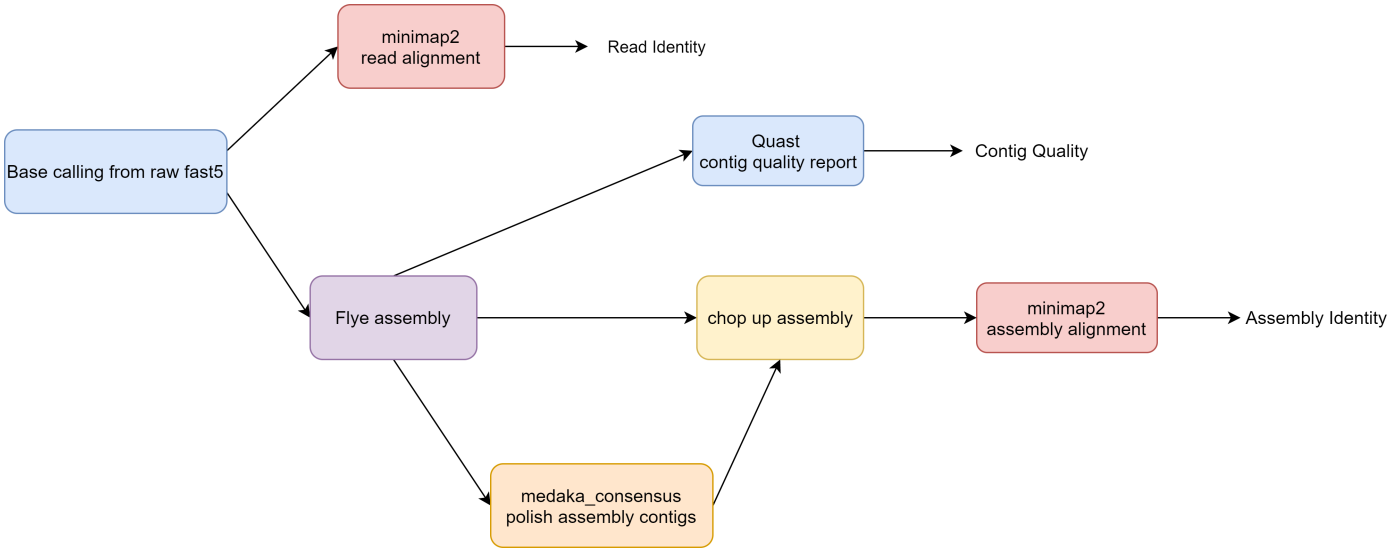


图 10: 基准测试分析流程

表 1: 基准测试数据集

Species	Sample	Reference chromosome size (bp)	GC-content	Flowcell type
Klebsiella pneumoniae	INF032	5,111,537	57.60%	R9.4
Klebsiella pneumoniae	INF042	5,337,491	57.40%	R9.4
Klebsiella pneumoniae	KSB2_1B	5,228,889	57.60%	R9.4
Klebsiella pneumoniae	NUH29	5,134,281	57.60%	R9.4
Shigella sonnei	2012−02037	4,829,160	51.00%	R9.4
Serratia marcescens	17−147−1671	5,517,578	59.10%	R9.4.1
Acinetobacter pittii	16−377−0801	3,814,719	38.80%	R9.4.1
Stenotrophomonas maltophilia	17_G_0092_Kos	4,802,733	66.30%	R9.4
Staphylococcus aureus	CAS38_02	2,902,076	32.80%	R9.4.1

碱基识别的准确率可以通过碱基序列与参考基因组的比对结果得到，计算公式如下：

$$identity = \frac{M}{M + S + I + D} * 100\%$$

其中，M 代表匹配的碱基数，S 代表不匹配的碱基数，I 代表插入的碱基数量，D 代表删除的碱基数量。测试结果如图 11所示，表格 2是识别准确率的中位数。可以看到在绝大多数测试集里，相比纳米孔公司的 Guppy 模型，SACall 的识别准确率更高，提升了 1 到 2 个百分比。改进的 SACall-Conv 相比原来的 SACall 准确率有了进一步的提升，提高了 0.2 到 0.5 个百分比。

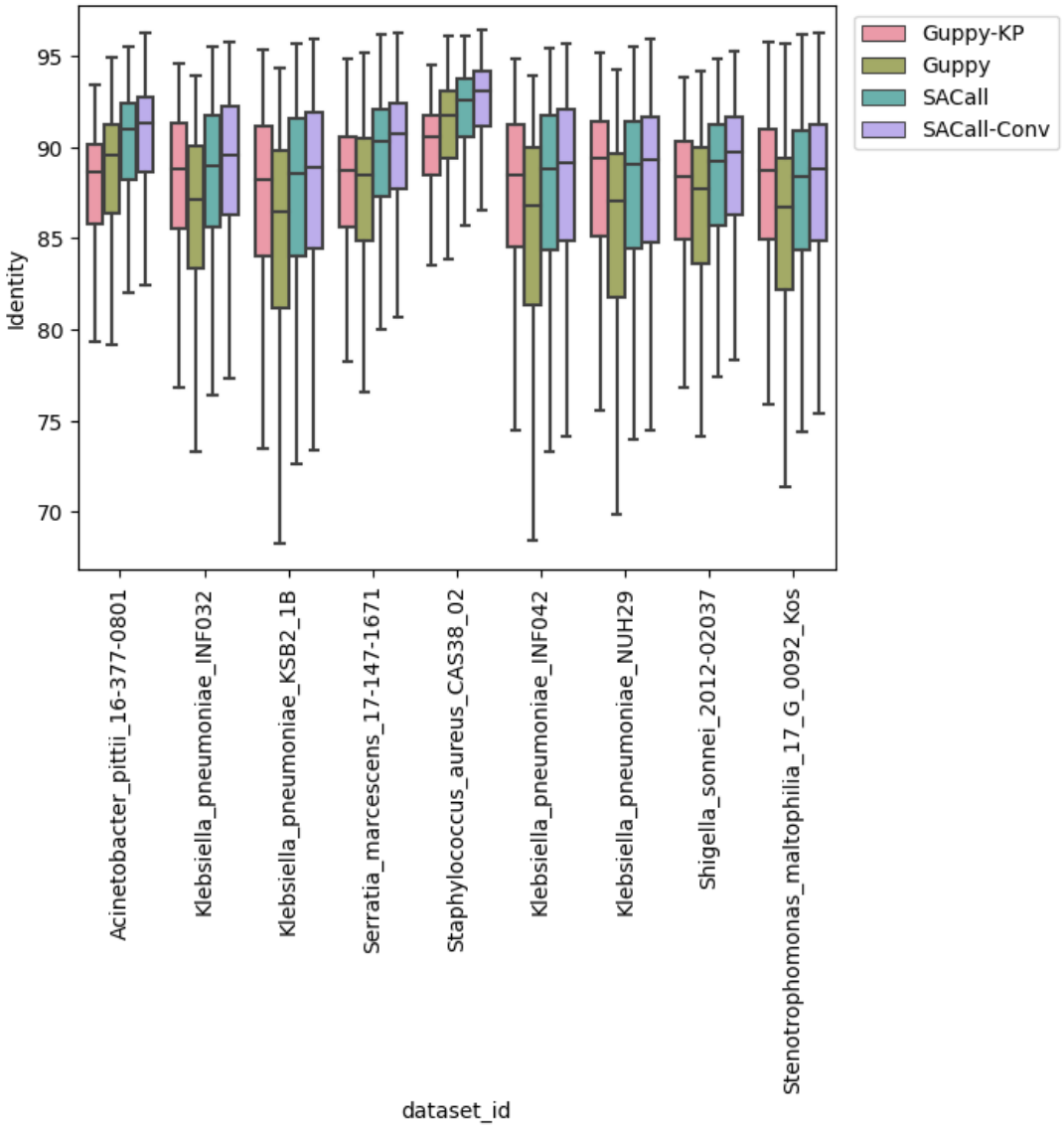


图 11: 碱基识别准确率的四分位图

表 2: 碱基识别准确率的中位数

Median read identity	Guppy-KP	Guppy	SACall	SACall-Conv
Acinetobacter_pittii_16-377-0801	88.65	89.57	91.01	91.36
Klebsiella_pneumoniae_INF032	88.78	87.17	89.01	89.57
Klebsiella_pneumoniae_KSB2_1B	88.25	86.48	88.55	88.91
Serratia_marcescens_17-147-1671	88.76	88.45	90.35	90.74
Staphylococcus_aureus_CAS38_02	90.55	91.76	92.60	93.07
Klebsiella_pneumoniae_INF042	88.50	86.78	88.79	89.13
Klebsiella_pneumoniae_NUH29	89.41	87.04	89.11	89.34
Shigella_sonnei_2012-02037	88.42	87.76	89.26	89.75
Stenotrophomonas_maltophilia_17_G_0092_Kos	88.74	86.68	88.42	88.79

组装的评估结果见表 3，可以看到在大部分数据集中，SACall-Conv 的组装准确率都是最高的，并且组装纠错以后准确率有了进一步的提升。在 N50、NA50、Genome fraction 这些指标上，不同模型之间基本是一致的。在每 100kbp 的不匹配碱基数指标上，SACall-Conv 更占优；而在每 100kbp 的插入删除碱基数指标里，Guppy 和 SACall-Conv 互有胜负。

表 3: 组装结果对比

dataset_id	model	# misassemblies	# mismatches per 100 kbp	# indels per 100 kbp	N50(Mbp)	NA50(Mbp)	Genome fraction (%)	Assembly identity(%)	Polished assembly identity(%)
Acinetobacter_pittii_16-377-0801	Guppy	0	25.19	229.89	3.8	3.8	99.975	99.75	99.85
	Guppy-KP	0	34.19	292.98	3.8	3.8	99.975	99.68	99.77
	SACall	0	12.21	323.29	3.8	3.8	99.975	99.66	99.83
	SACall-Conv	0	6.99	266.98	3.8	3.8	99.975	99.72	99.86
Klebsiella_pneumoniae_INF032	Guppy	0	389.37	383.97	5.1	5.1	100.000	99.21	99.47
	Guppy-KP	0	17.25	237.96	5.1	5.1	100.000	99.74	99.92
	SACall	0	22.20	425.52	5.1	5.1	100.000	99.51	99.85
	SACall-Conv	0	16.66	256.60	5.1	5.1	100.000	99.72	99.90
Klebsiella_pneumoniae_INF042	Guppy	0	207.22	266.90	5.3	5.3	100.000	99.51	99.71
	Guppy-KP	0	17.17	253.82	5.3	5.3	100.000	99.72	99.93
	SACall	0	9.52	403.62	5.3	5.3	100.000	99.55	99.87
	SACall-Conv	0	5.73	225.94	5.3	5.3	100.000	99.76	99.94
Klebsiella_pneumoniae_KSB2_1B	Guppy	0	416.16	299.84	5.2	5.2	100.000	99.27	99.46
	Guppy-KP	0	22.07	253.37	5.2	5.2	100.000	99.72	99.92
	SACall	0	19.21	400.16	5.2	5.2	100.000	99.54	99.85
	SACall-Conv	0	11.92	229.08	5.2	5.2	99.074	99.75	99.92
Klebsiella_pneumoniae_NUH29	Guppy	0	383.18	208.20	5.1	5.1	100.000	99.39	99.41
	Guppy-KP	0	16.67	228.97	5.1	5.1	100.000	99.74	99.89
	SACall	0	10.15	347.32	5.1	5.1	100.000	99.61	99.84
	SACall-Conv	0	8.16	217.16	5.1	5.1	100.000	99.76	99.90
Serratia_marcescens_17-147-1671	Guppy	2	64.24	238.08	5.5	5.5	99.959	99.70	99.78
	Guppy-KP	1	93.98	317.90	5.5	5.4	99.244	99.59	99.72
	SACall	0	43.89	304.39	5.5	5.5	99.990	99.63	99.80
	SACall-Conv	1	34.59	206.13	5.5	5.5	99.971	99.76	99.86
Shigella_sonnei_2012-02037	Guppy	1	274.53	445.05	4.8	2.5	99.946	99.29	99.60
	Guppy-KP	0	21.68	259.40	4.8	4.8	99.971	99.71	99.92
	SACall	1	25.25	491.75	4.8	2.5	99.946	99.46	99.83
	SACall-Conv	0	17.21	263.78	4.8	4.8	99.970	99.73	99.92
Staphylococcus_aureus_CAS38_02	Guppy	0	10.77	220.60	2.9	2.9	100.000	99.78	99.96
	Guppy-KP	0	14.08	187.44	2.9	2.9	100.000	99.81	99.93
	SACall	0	9.13	299.19	2.9	2.9	100.000	99.69	99.89
	SACall-Conv	0	6.80	244.80	2.9	2.9	100.000	99.75	99.94
Stenotrophomonas_maltophilia_17_G_0092_Kos	Guppy	0	12.67	78.59	4.8	4.8	100.000	99.90	99.93
	Guppy-KP	0	24.46	163.30	4.8	4.8	100.000	99.80	99.93
	SACall	0	13.84	214.04	4.8	4.8	100.000	99.75	99.94
	SACall-Conv	0	11.94	104.07	4.8	4.8	100.000	99.88	99.95

6 总结与展望

本次课程的论文复现并改进了基于自注意力机制的纳米孔碱基测序信号识别模型，在复现的过程中，通过实验验证和分析取得了相对初步的结果。不足之处在于，受限于计算资源的限制，我们还没有尝试其他的改进方案，并且模型的推理性能还有较多的改进空间。未来我们会尝试在现有的框架上，对模型进行更多的尝试，并且尽可能减少计算开销，以提高整个识别程序的效率。

参考文献

[1] MATTHIJS G, SOUCHE E, ALDERS M, et al. Guidelines for diagnostic next-generation sequencing [J]. European Journal of Human Genetics, 2016, 24(1): 2-5.

[2] JAIN M, OLSEN H E, PATEN B, et al. The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community[J]. Genome biology, 2016, 17(1): 1-11.

[3] GUO R, LI Y R, HE S, et al. RepLong: de novo repeat identification using long read sequencing data [J]. Bioinformatics, 2018, 34(7): 1099-1107.

[4] SIMPSON J T, WORKMAN R E, ZUZARTE P, et al. Detecting DNA cytosine methylation using nanopore sequencing[J]. Nature methods, 2017, 14(4): 407-410.

[5] LOOSE M, MALLA S, STOUT M. Real-time selective sequencing using nanopore technology[J]. Nature methods, 2016, 13(9): 751-754.

[6] DAVID M, DURSI L J, YAO D, et al. Nanocall: an open source basecaller for Oxford Nanopore sequencing data[J]. Bioinformatics, 2017, 33(1): 49-55.

[7] BOŽA V, BREJOVÁ B, VINAŘ T. DeepNano: deep recurrent neural networks for base calling in MinION nanopore reads[J]. PloS one, 2017, 12(6): e0178751.

[8] SCHUSTER M, PALIWAL K K. Bidirectional recurrent neural networks[J]. IEEE transactions on Signal Processing, 1997, 45(11): 2673-2681.

[9] CHUNG J, GULCEHRE C, CHO K, et al. Empirical evaluation of gated recurrent neural networks on sequence modeling[J]. arXiv preprint arXiv:1412.3555, 2014.

[10] TENG H, CAO M D, HALL M B, et al. Chiron: translating nanopore raw signal directly into nucleotide sequence using deep learning[J]. GigaScience, 2018, 7(5): giy037.

[11] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[12] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[13] WICK R R, JUDD L M, HOLT K E. Performance of neural network basecalling tools for Oxford Nanopore sequencing[J]. Genome biology, 2019, 20(1): 1-10.

[14] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

[15] GRAVES A, FERNÁNDEZ S, GOMEZ F, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks[C]//Proceedings of the 23rd international conference on Machine learning. 2006: 369-376.

[16] SANTURKAR S, TSIPRAS D, ILYAS A, et al. How does batch normalization help optimization?[J]. Advances in neural information processing systems, 2018, 31.

[17] AGARAP A F. Deep learning using rectified linear units (relu)[J]. arXiv preprint arXiv:1803.08375, 2018.

[18] LI H. Minimap2: pairwise alignment for nucleotide sequences[J]. Bioinformatics, 2018, 34(18): 3094-3100.

[19] KOLMOGOROV M, YUAN J, LIN Y, et al. Assembly of long, error-prone reads using repeat graphs [J]. Nature biotechnology, 2019, 37(5): 540-546.

[20] GUREVICH A, SAVELIEV V, VYAHHI N, et al. QUAST: quality assessment tool for genome assemblies[J]. Bioinformatics, 2013, 29(8): 1072-1075.