**基于机器学习的开源项目生存状态预测技术研究和基准数据集构建**

**1. 研究背景与立项依据**

开源软件是通过开放协作开发和维护的软件，通常免费提供给任何人使用、检查、修改和重新分发。这与专有或封闭源代码软件应用程序（如 Microsoft Word、 Adobe Illustrator）形成鲜明对比，这些应用程序由创作者或版权所有者出售给最终用户，除非版权所有者另有规定，否则不得进行编辑、增强或重新发布。“开源”一词还更广泛地指基于社区的方法，通过开放协作、包容性、透明度和频繁的公开更新来创建任何知识产权（如软件）。开源是一个最初指开源软件（OSS）的术语。开源软件是设计为任何人都可以看到、修改和分发他们认为合适的代码的公开访问的代码。开源软件是以分散和协作的方式开发的，依赖于同行评审和社区生产。开源软件通常比其专有软件更便宜、更灵活、寿命更长，因为它是由社区而不是单个作者或公司开发的。开源已经成为一种超越软件生产的运动和工作方式。开源运动利用开源软件的价值观和去中心化生产模式，寻找解决社区和行业问题的新方法。

开源软件虽然数量庞大，但是成功率不高，很多项目在一年内就放弃开发进入沉寂的状态。对于开源软件的用户、希望参与开源软件的开发者、希望投资开源软件的企业和基金组织而言，如何预判项目未来的持续活跃，或者预警可能的沉寂风险，就成为一个重要的问题。普通用户会更希望加入成功概率高、组织较为规范的项目，而不是在无序的项目中浪费时间。与此同时，企业和基金组织也会希望通过一定的科学评估方法来判断项目的投资潜力，从而对比较有前景的项目进行持续投入。而作为项目的管理者，通过一定的方法持续监控项目的状态也是非常有必要的。

目前，基于各类指标和机器学习算法的开源软件项目活跃度预测，展现出了一定的有效性，有望给用户选择健康的项目提出有效建议。一些学者通过生存模型[2]对项目的活跃度/成败概率进行建模和预测，一些学者[10]通过对用户的“turn over”建模来进行生存预测，也有相关工作探讨了使用机器学习模型来对项目的未来活跃程度进行预测的方法[14,15]。正如相关工作中所总结的，现有工作提供了一系列不同的指标、预测方法，并且基于不同的数据集开展了研究并取得了良好的测试结果。但是难以相互之间进行有效对比。

针对上述现状，本项目拟提供一个较大规模的基准测试集，其中包含开源项目健康和活跃相关的度量指标、多种项目健康和活跃预测基准方法、全面的评价指标以及大规模的数据集。为通过量化的、面向开源社区的度量指标，基于机器学习和人工智能方法预测项目的活跃和成败提供支持。

**2. 相关工作**

围绕开源项目及其社区的生存度量和分析，现有工作开展了丰富的研究。

**2.1. 开源软件社区健康度量**

在[6]中， Linåker 等人使用滚雪球式的方法开展文献调研（snowballing），总结了描述开源软件项目健康的特征，他们给出的评估总结包括 15 个主题的 107 个特征，主要包括以下几类特征:（1）以人为核心的特征（2）以软件为核心的特征（3）以协调为核心的特征。这篇工作包括与总结了当前主流的有关开源软件项目健康的内容。但是，该文的总结仅仅是对现有工作进行分类和介绍，没有对该领域给出结论性和指导性的意见。

在[7]中， Yousef 等人给出了有关软件生态系统健康度量和方法的分类与总结，给出了有关软件生态系统健康的 7 种不同的定义， 142 个度量值和 174 种不同的度量方法。该综述中给出了三个维度的分类，包括： 利基创造力（niche creation）、 生产力（productivity）和稳健性（robustness），并详细阐述了这三类分类中包括的具体度量指标和方法。类似地，在[11]中， Alves 等学者针对软件生态系统中的治理机制及其健康这一主题开展了系统性的文献综述。通过对其所调研的相关文献进行梳理和提炼， Alves 等总结一个软件生态系统的健康可以由一个生态系统持续产出有意义产品的能力（即，生产力）、能够在市场波动中存活下来（即，稳健性）、在生态系统中产生创新（即，利基创造力）这三个方面决定。围绕这三个方面， Alves 等分类梳理了共 52 个具体的度量软件生态系统健康的指标，形成了相应的指标体系。

在[8]中，Van 等人以软件生态系统的可持续性评估为主题，调研和总结了当前领域的主要成果。本文主要通过问卷和调查的方式完成对如下问题的回答：（1）可持续性是如何影响软件生态系统的？（2）当前有哪些方法可以用于分析软件生态系统的可持续性？它们有什么缺陷？（3）软件生态系统工业界和学术界是如何看待软件生态系统的可持续性？（4）软件生态系统的可持续性话题主要有哪些？（5）软件生态系统的可持续性分析方法在学术界和工业界的应用的阻碍在哪里？最终得出的结论为，软件生态系统的可持续性作为一个新概念，还未在业界广泛使用和评估，但是调研的结果表示，其潜在价值值得引起重视。

在[9]中， Sean 等人讨论了有关开源社区健康的分析度量指标以及它们具体的操作定义，总结了度量开源社区的成功度、 可持续性、 风险、 健康四个主题的指标。该文同时给出了在CHAOSS 社区中这些有关开源社区健康的指标是如何计算和展示的，可以发现该类指标可以很好地展示开源社区的健康程度。  
 在[10]中， Liao 等人也对开源生态系统的可持续性评估模型和方法给出综述。该文总结评估系统为三个层面： 目标层面、 指导层面和评估层面；同时将评估指标分为以下几个类别： 开放性、 稳定性、 活跃性和可拓展性。在已有的度量指标和系统的基础上，该文提出了结合以上所有内容的加权计算评估方式。  
 在[12-13]中， Manikas 与 Hansen 针对软件生态系统的健康，对相关工作进行了综述并提出了一个概念框架。该概念框架参照[13]将软件生态系统的健康指标分类为人（actors）、软件（software）和调度协（orchestration）三个侧面。

通过对现有工作进行总结，我们发现现有研究工作为我们度量开源软件及其社区的健康提供了丰富的指标体系，为我们进一步开展基于机器学习的开源项目生存状态预测提供了依据。但同时我们发现，许多相关研究仍停留在指标设计和小规模测试的阶段，目前尚缺乏一套具有较大规模的基准数据集可供使用，需要我们开展工作。

**2.2. 项目的活跃和生存预测**

在研究开源社区项目survivalability的工作中，常见的数据切分方法是认为”一旦没有任何修改，项目就会死亡“[2-5]。此外，也有人认为人认为一个项目如果一个月收到的修订少于两次，则该项目是不活跃的；两个月的不活动导致其被认为是被遗弃或死亡。

对于项目的活跃和生存预测，目前的主要研究方法有基于回归分析的方法进行预测、基于生存分析的方法进行预测和基于机器学习方法进行预测。Raja[18]提出了一个VI指标来分析项目的幸存概率，他认为项目生存能力有三个维度：活力、韧性和组织，并定义了这些维度中的每一个维度，并制定了一个名为生存能力指数( VI )的指数来组合所有的三个维度。还有一些学者[2-5]借鉴医学中的生存分析方法对项目的生存概率进行预测分析，并通过KM估计值来计算生存概率的具体值。此外，目前有很多研究[19]中将机器学习方法用于生存性分析，用于处理此类问题中的删失数据，并解决实际数据中出现的其他挑战性问题。

在这个问题中我们试图通过引入机器学习和时间序列分类预测技术进行项目生存状态预测。在生存性分析的机器学习方法[19]中，一方面他们并不特定针对开源社区项目的具体应用场景，另一方面他们主要针对生存性分析中的统计指标进行改进，与我们想要的预测功能存在一定差异。我们试图通过这个工作来填补使用机器学习方法进行项目生存预测的空白。

**2.3. 时间序列分类预测方法**

上面描述了项目的活跃与生存预测，进一步讲，我们试图通过结合时间序列分类预测方法来实现这一目标。一般来讲，基于距离的时间序列分类[1]如下：

* 基于特征的：在基于特征的分类方法中，时间序列被转化为特征向量，然后通过传统的分类器如神经网络或决策树进行分类。一些用于特征提取的方法包括谱方法，如离散傅里叶变换( DFT )或离散小波变换( DWT )，考虑频域特征的，或奇异值分解( SVD )，奇异值分解( SVD )，进行特征值分析，以减少特征集的同时保留相关信息。
* 基于模型的：基于模型的分类假设一个类别中的所有时间序列都是由相同的底层模型生成的，从而将一个新的序列分配给最符合该模型的类别。一些基于模型的方法是利用自回归模型[20]或隐马尔可夫模型[21]等形成的。
* 基于距离的：基于距离的方法是定义序列之间的(或非)相似性度量，然后在基于距离的分类方法中，如k -最近邻分类器( k-NN )或支持向量机( SVMs )中，以某种方式引入这些距离。

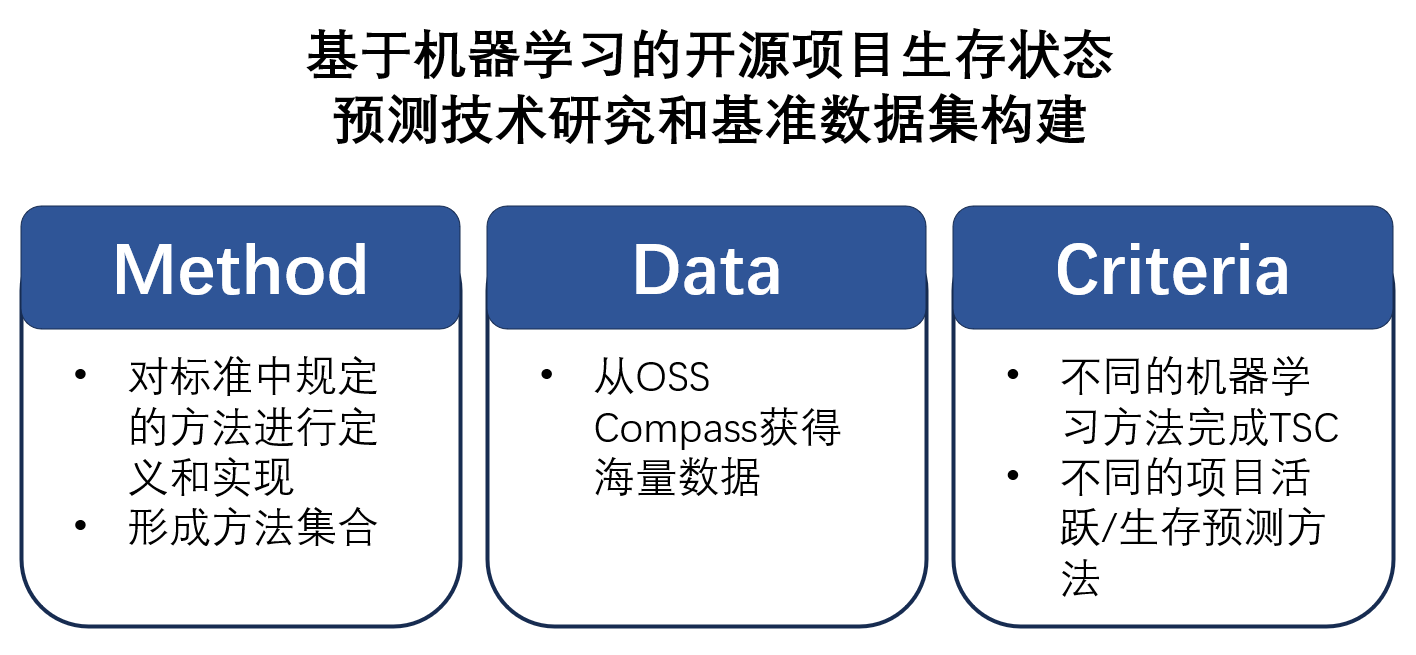
由于基于模型的时间序列分类方法并不适用于本项目的应用场景，因此我们将基于模型的时间序列分类改为基于深度学习的相关方法进行预测。此外，在距离的预测方法中，我们选择通过shapelet模式挖掘来进行预测。

对于深度学习相关方法，目前已有很多种DNN，用于TSC任务的主要有三种：

* 多层感知器MLP：一个MLP构成了深度学习模型最简单、最传统的架构。这种结构形式也被称为全连接( FC )网络，因为li层的神经元与li - 1层的每个神经元相连，i∈[ 1 , L]。
* 卷积神经网络CNN：自AlexNet [22]在2012年获得ImageNet大赛冠军以来，深度卷积神经网络已经在许多不同的领域[23]中得到了成功的应用，例如在图像识别问题[24]中达到人类水平的性能，以及不同的自然语言处理任务[25,26]。受这些CNN架构在这些不同领域的成功启发，研究人员开始将其用于时间序列分析[27]。卷积可以看作是在时间序列上施加和滑动一个滤波器。与图像不同的是，滤波器只显示一个维度(时间)而不是两个维度(宽度和高度)。该滤波器也可以看作是时间序列的一般非线性变换。
* 回声状态网络ESN：另一种流行的深度学习模型架构是循环神经网络( Recurrent Neural Network，RNN )。除了时间序列预测，我们发现这些神经网络很少应用于时间序列分类，这主要是由于三个因素：( 1 )这种架构的类型主要是为了预测时间序列中每个元素(时间戳)的输出；( 2 )由于在长时间序列上进行训练，RNNs通常会面临梯度消失问题；( 3 ) RNNs被认为难以训练和并行化，导致研究人员出于的计算原因而避免使用它们。鉴于上述局限性，针对时间序列提出了一种相对较新的循环结构：回声状态网络( Echo State Networks，ESNs )[28]。它们的设计是为了缓解RNNs的挑战，消除了需要计算隐藏层梯度的问题，从而减少了这些神经网络的训练时间，从而避免了梯度消失问题。

**3. 研究内容和技术路线**

在大规模数据集上，基于设计好的标准，实现一系列具体方法，完成下列流程：



**3.1 数据准备**

拟从GitHub和Gitee上选取部分（例如5000个）仓库，形成一个仓库集合R。根据R从OSS Compass获取预测需要的全部数据（与社区演化相关的行为数据）。并且将获取到的数据以结构化的形式组织到数据库中进行存储和访问。

**3.2 “终止事件”**

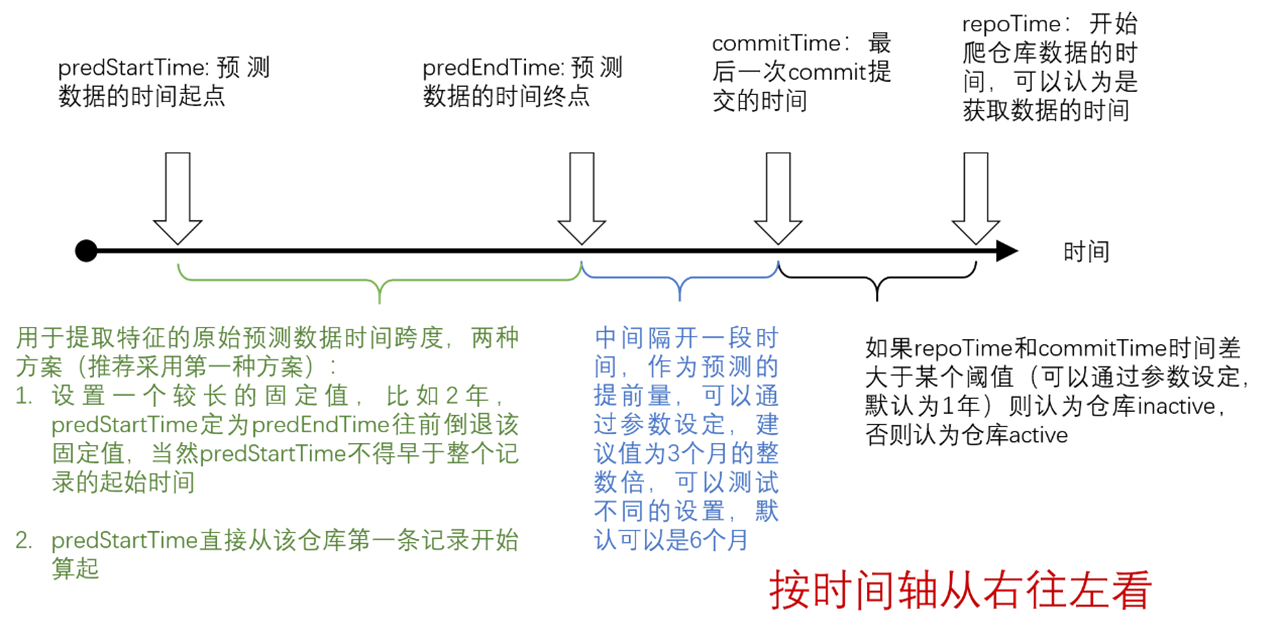
在以项目为粒度获取数据以后，需要对每个仓库（也称为样本）打标签，判断他们在当前状态下是否为活跃项目。尽管很多研究提出[3,4]，由于项目可能会在长时间停滞后被重新开发，也就是“起死回生”，因此无法通过对项目的历史数据观察完全得出项目是否成功/失败的结论，但是判断项目是否活跃/维护（active/maintained），可以在很大程度上预示项目的成功和失败。

判断项目是否活跃的一个常见方法是借鉴幸存模型的定义，识别时间序列中是否有“终止事件”的发生，“终止事件”的发生代表着项目幸存能力的下降。在本问题中，“终止事件”指的是“是否无commit的提交”，在开源社区中，项目代码贡献通过commit提交到仓库中，因此视此为识别项目活跃度的标志。

**3.3 数据预处理和分割方法**

在这一步，原始的行为数据被社区演化算法计算转化为四个指数的多维时间序列。在这部分中，涉及到数据的切割和重新组织：

* 首先是将原始数据序列切割成合适的长度，截取其中的有效数据。一般做法是切割最近的两年有效数据，并且以一年为时间长度确定项目是否判定为终止[2,4]。在我们的问题中，为了保证灵活性，应该将时间长度、提前量等设置为参数。
* 在计算四个指数的多维时间序列时，需要将数据按照时间顺序组织并切片[16]。例如：时间切片长度为一个月，每次向前滑动一周，将2023.04-2023.06的数据切分成4个切片。



**3.4 机器学习方法、验证方法**

这部分拟使用不同的机器学习方法进行上述的时间序列预测，将方法分为基于特征的预测方法、基于距离的预测方法、基于深度学习的预测方法三类。

* 基于特征的预测方法：这部分主要指针对多维时间序列，进行与时间序列相关的特征挖掘[17]，并将挖掘出来的特征作为输入，进行机器学习模型的训练和测试。
* 基于距离的预测方法（模式挖掘）：这部分主要指基于shapelet的模式挖掘方法，进行的预测。shapelet方法是基于距离矩阵的，因此距离计算结果会对结果产生影响，同时也常用一些基于距离的预测方法进行分类和预测[1]。例如使用KNN、距离内核等方法。
* 基于深度学习模型的预测方法：对于时间序列分类问题，目前比较新的方法是采用深度学习模型（如CNN等）进行分类[14]。这类方法需要更多的运算资源，但是往往能够取得更高的精确度和分类效果。

在经过上述步骤得到一系列分类模型以后，就可以对模型的效果进行评估。常见的方法是采用十折交叉验证，这种方法可以得到较为精确的评估结果，避免因数据集不同切分方式导致的误差。

**3.5 评价指标体系**

这一步拟对测试评估结果进行整理、比较、分析、讨论、总结，主要是比较不同方法之间的精确度、混淆矩阵等评估指标的差异，分析每种方法不同的优缺点和适用范围，并试图得到一些值得探讨的结论。

**参考文献**

[1] Amaia Abanda, Usue Mori, and Jose A. Lozano. 2019. A Review on Distance Based Time Series Classification. Data Mining and Knowledge Discovery 33, 2 (March 2019), 378–412. https://doi.org/10.1007/s10618-018-0596-4  
[2] Rao Hamza Ali, Chelsea Parlett-Pelleriti, and Erik Linstead. 2020. Cheating Death: A Statistical Survival Analysis of Publicly Available Python Projects. In Proceedings of the 17th International Conference on Mining Software Repositories. ACM, Seoul Republic of Korea, 6–10. https://doi.org/10.1145/3379597.3387511  
[3] Jailton Coelho and Marco Tulio Valente. 2017. Why modern open source projects fail. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. 186–196.  
[4] Jailton Coelho, Marco Tulio Valente, Luciano Milen, and Luciana L. Silva. 2020. Is This GitHub Project Maintained? Measuring the Level of Maintenance Activity of Open-Source Projects. Information and Software Technology 122 (June 2020), 106274. https://doi.org/10.1016/j.infsof.2020.106274  
[5] Nicholas Evangelopoulos, Anna Sidorova, Stergios Fotopoulos, and Indushobha Chengalur-Smith. 2008. Determining Process Death Based on Censored Activity Data. Communications in Statistics - Simulation and Computation 37, 8 (Aug. 2008), 1647–1662. https://doi.org/10.1080/03610910802140224

[6] LINÅKER J, PAPATHEOCHAROUS E, OLSSON T. How to characterize the health of an open source software project? a snowball literature review of an emerging practice[C]// Proceedings of the 18th International Symposium on Open Collaboration. [S.l.: s.n.], 2022: 1-12.  
[7] YOUSEF ZADEH SHOOSHTARI A. Taxonomies of software ecosystem health metrics and practices: a systematic literature review[D]. [S.l.: s.n.], 2020.  
[8] VAN DER SLUIJS D. Software ecosystem sustainability assessment: an investigation of the state of the art and stakeholder perceptions[J]. 2018.  
[9] GOGGINS S, LUMBARD K, GERMONPREZ M. Open source community health: Analytical metrics and their corresponding narratives[C]//2021 IEEE/ACM 4th International Workshop on Software Health in Projects, Ecosystems and Communities (SoHeal). [S.l.]: IEEE, 2021: 25-33.  
[10] LIAO Z, DENG L, FAN X, et al. Empirical research on the evaluation model and method of sustainability of the open source ecosystem[J]. Symmetry, 2018, 10(12):747.

[10] Lin, Bin, Gregorio Robles和Alexander Serebrenik. 《Developer Turnover in Global, Industrial Open Source Projects: Insights from Applying Survival Analysis》. 收入 *2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)*, 66–75. Buenos Aires, Argentina: IEEE, 2017. <https://doi.org/10.1109/ICGSE.2017.11>.

[11] ALVES C F, OLIVEIRA J A, JANSEN S. Understanding governance mechanisms and health in software ecosystems: A systematic literature review[C]//International Conference on Enterprise Information Systems. [S.l.: s.n.], 2017.  
[12] MANIKAS K, HANSEN K M. Reviewing the health of software ecosystems–a conceptual framework proposal[C]//Proceedings of the 5th international workshop on software ecosystems (IWSECO). [S.l.]: Citeseer, 2013: 33-44.  
[13] MANIKAS K, HANSEN K M. Software ecosystems - a systematic literature review[J]. J. Syst. Softw., 2013, 86:1294-1306.

[14] Ismail Fawaz, Hassan, Germain Forestier, Jonathan Weber, Lhassane Idoumghar和Pierre-Alain Muller. 《Deep Learning for Time Series Classification: A Review》. *Data Mining and Knowledge Discovery* 33, 期 4 (2019年7月1日): 917–63. <https://doi.org/10.1007/s10618-019-00619-1>.

[15] Karim, Fazle, Somshubra Majumdar, Houshang Darabi和Samuel Harford. 《Multivariate LSTM-FCNs for Time Series Classification》. *Neural Networks* 116 (2019年8月): 237–45. <https://doi.org/10.1016/j.neunet.2019.04.014>.

[16] Liang Wang, Ying Li, Jierui Zhang, and Xianping Tao. 2022. Quantifying community evolution in developer social networks. In Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2022). Association for Computing Machinery, New York, NY, USA, 157–169. https://doi.org/10.1145/3540250.3549106

[17] Zhang, J., Wang, L., Zheng, Z., Tao, X. (2022). Social Community Evolution Analysis and Visualization in Open Source Software Projects. In: Chbeir, R., Huang, H., Silvestri, F., Manolopoulos, Y., Zhang, Y. (eds) Web Information Systems Engineering – WISE 2022. WISE 2022. Lecture Notes in Computer Science, vol 13724. Springer, Cham. https://doi.org/10.1007/978-3-031-20891-1\_4

[18] Raja, Uzma, 和Marietta J. Tretter. 《Defining and Evaluating a Measure of Open Source Project Survivability》. *IEEE Transactions on Software Engineering* 38, 期 1 (2012年1月): 163–74. <https://doi.org/10.1109/TSE.2011.39>.

[19] Wang, Ping, Yan Li和Chandan K. Reddy. 《Machine learning for survival analysis: A survey》. *Acm Computing Surveys* 51, 期 6 (2019年2月). <https://doi.org/10.1145/3214306>.

[20] Bagnall A, Janacek G (2014) A run length transformation for discriminating between autoregressive time series. J Classif 31(2):274–295

[21] Smyth P (1997) Clustering sequences with hidden Markov models. Adv Neural Inf Process Syst 9:648–654 Sun R, Luo ZQ (2016) Guaranteed matrix completion via non-convex factorization. IEEE Trans Inf Theory 62(11):6535–6579

[22] Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Advances in neural information processing systems vol 25, pp 1097–1105

[23] LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444

[24] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: IEEE conference on computer vision and pattern recognition, pp 1–9

[25] Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Neural information processing systems, pp 3104–3112

[26] Bahdanau D, Cho K, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: International conference on learning representations

[27] Cristian Borges Gamboa J (2017) Deep learning for time-series analysis. arXiv:1701.01887

[28] Gallicchio C, Micheli A (2017) Deep echo state network (DeepESN): a brief survey. arXiv:1712.04323