Journal of Machine Learning Research 21 (2021) 1-7 Submitted 01/21; Revised ??/21; Published ??/21

PyHealth: A Python Library for Health Predictive Models

Yue Zhao* ZHAOY@CMU.EDU
Carnegie Mellon University†

Zhi Qiao* ZHI.QIAO@IQVIA.COM
Analytics Center of Excellence, IQVIA

Cao Xiao CAO.XIAO@IQVIA.COM
Analytics Center of Excellence, IQVIA

Lucas M. Glass LUCAS.GLASS@IQVIA.COM
Analytics Center of Excellence, IQVIA

Jimeng Sun JIMENG@ILLINOIS.EDU
University of Illinois Urbana-Champaign

Editor: xxx xxx

Abstract

Despite the explosion of interest in healthcare AI research, the reproducibility and benchmarking of those research works are often limited due to the lack of standard benchmark datasets and diverse evaluation metrics. To address this reproducibility challenge, we develop PyHealth , an open-source Python toolbox for developing various predictive models on healthcare data.

PyHealth consists of data preprocessing module, predictive modeling module, and evaluation module. The target users of PyHealth are both computer science researchers and healthcare data scientists. With PyHealth , they can conduct complex machine learning pipelines on healthcare datasets with fewer than ten lines of code. The data preprocessing module enables the transformation of complex healthcare datasets such as longitudinal electronic health records, medical images, continuous signals (e.g., electrocardiogram), and clinical notes into machine learning friendly formats. The predictive modeling module provides more than 30 machine learning models, including established ensemble trees and deep neural network-based approaches, via a unified but extendable API designed for both researchers and practitioners. The evaluation module provides various evaluation strategies (e.g., cross-validation and train-validation-test split) and predictive model metrics.

With robustness and scalability in mind, best practices such as unit testing, continuous integration, code coverage, and interactive examples are introduced in the library's development. PyHealth can be installed through the Python Package Index (PyPI) or `https://github.com/yzhao062/PyHealth`.

Keywords: healthcare, deep learning, neural networks, machine learning, Python

∗. Y. Zhao & Z. Qiao contributed equally to this work.
†. Work initialized while at IQVIA.

1. Introduction

With the advances of machine learning technologies and the tremendous improvements in digital healthcare systems, digitized healthcare data have demonstrated initial success in enhancing health decision making and healthcare delivery (Xiao et al., 2018). However, health data are usually noisy, complex, and have heterogeneous forms, yielding a wide range of healthcare modeling tasks. For example, health risk prediction based on sequential patient data (Gao et al., 2020a), medical image-based disease diagnosis (Qiao et al., 2020), risk detection based on continuous physiological signals. e.g., electroencephalogram (EEG) or electrocardiogram (ECG) and multimodal clinical notes (e.g., text and images). The complexity and heterogeneity of health data and tasks lead to the long overdue of a dedicated ML system for benchmarking predictive health models despite their high value in healthcare research and clinical decision making.

Several efforts have been made to help approach these problems, including releasing benchmark urgent care datasets MIMIC-III (Harutyunyan et al., 2019), streamlining the analytic pipeline (Miller et al., 2017). However, a dedicated toolkit for predictive health is absent. To fill the gap, we design and implement PyHealth —a comprehensive Python library that provides a full-stack machine learning toolkit for predictive health tasks, including data preprocessing training and prediction, and result evaluation.

PyHealth has five distinct advantages. First, it encapsulates more than 30 state-of-the-art predictive health algorithms, including both classical techniques such as XGBoost (Chen and Guestrin, 2016) and recent deep learning architectures such as autoencoders, convolution based, and adversarial based models (see Table 1). Second, PyHealth has a wide coverage and contains models for different data types, e.g., sequence data, image data, physiological signal data, and unstructured text data. Third, PyHealth includes a unified API, detailed documentation, and interactive examples across all algorithms for clarity and ease of use—executing complex deep learning models needs fewer than ten lines of code. Fourth, most models in PyHealth are covered by unit testing with cross-platform, continuous integration, code coverage, and code maintainability checks. Last, parallelization is enabled in select modules (data preprocessing) for efficiency and scalability, along with fast GPU computation for deel learning models through `PyTorch`.

2. Library Design and Implementation

PyHealth is designed for Python 3 and relies on `numpy`, `scipy`, `scikit-learn` and `PyTorch`. As shown in Fig. 1, PyHealth comes with three major modules: (i) ***data preprocessing module*** can take user input and validate and convert the input data into the format that learning models can easily handle; (ii) ***predictive modeling module*** comprises a suite of models that are grouped by input data type into *sequence*s, *image*, *EEG*, and *text*. For each data type, a group of dedicated learning models are implemented to be easily used, and (iii) ***evaluation module*** can infer the task type automatically, e.g., multi-classification, and conduct the comprehensive evaluation by task type.

Inspired by `scikit-learn`'s API design and general deep learning design, most learning models inherit from the corresponding base class with the same interface: (i) `fit` processes the train data and validation data to learn the weights and save necessary statistics; (ii)

Table 1: More than 30 Healthcare AI Models are implemented in PyHealth 0.0.6

Method	Sequence	Image	Signal	Text	Deep Learning
RandomForest (Breiman, 2001)	✓	✓	✓	✓	✗
XGBoost (Chen and Guestrin, 2016)	✓	✓	✓	✓	✗
BasicCNN (Lawrence et al., 1997)	✗	✓	✓	✓	✓
LSTM (Hochreiter and Schmidhuber, 1997)	✓	✗	✗	✗	✓
GRU (Cho et al., 2014)	✓	✗	✗	✗	✓
RETAIN (Choi et al., 2016)	✓	✗	✗	✗	✓
Dipole (Ma et al., 2017)	✓	✗	✗	✗	✓
tLSTM (Baytas et al., 2017)	✓	✗	✗	✗	✓
RAIM (Xu et al., 2018)	✓	✗	✗	✗	✓
StageNet (Gao et al., 2020b)	✓	✗	✗	✗	✓
Vggnet (Simonyan and Zisserman, 2014)	✗	✓	✗	✗	✓
Inception (Szegedy et al., 2016)	✗	✓	✗	✗	✓
Resnet (He et al., 2016)	✗	✓	✗	✗	✓
Resnext(Xie et al., 2016)	✗	✓	✗	✗	✓
Densenet (Huang et al., 2017)	✗	✓	✗	✗	✓
Mobilenet (Howard et al., 2017)	✗	✓	✗	✗	✓
DBLSTM-WS (Özal Yildirim, 2018)	✗	✗	✓	✗	✓
DenseConv (Parvaneh et al., 2018)	✗	✗	✓	✗	✓
DeepRes1D (Li et al., 2019)	✗	✗	✓	✗	✓
Autoencoder+BiLSTM (Wang et al., 2019)	✗	✗	✓	✗	✓
KRCRnet (Zhou et al., 2019)	✗	✗	✓	✗	✓
MINA (Hong et al., 2019)	✗	✗	✓	✗	✓
Bidirectional-GRU (Cho et al., 2014)	✗	✗	✗	✓	✓
DR-CAML (Mullenbach et al., 2018)	✗	✗	✗	✓	✓
DCAN (Ji et al., 2020)	✗	✗	✗	✓	✓
JointLAAT (Vu et al., 2020)	✗	✗	✗	✓	✓
MultiResCNN (Li and Yu, 2020)	✗	✗	✗	✓	✓

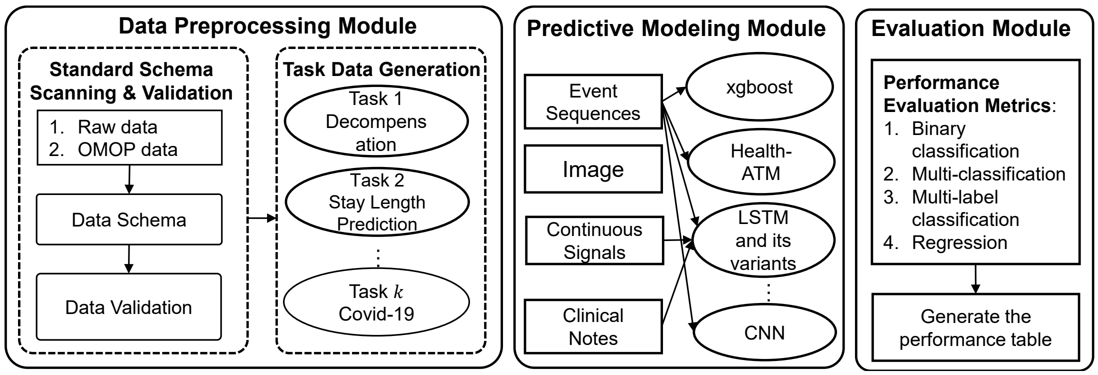
`load_model` selects the model with the highest validation accuracy and (iii) `inference` takes in incoming test data and make a prediction on it. A short demo of using these APIs is presented below. Other tasks follow a similar API.

```
>>> from pyhealth.models.sequence.lstm import LSTM
>>> from pyhealth.evaluation.evaluator import func

>>> current_data = expdata_generator(exp_id=exp_id) # initialize dataset
>>> current_data.get_exp_data(sel_task="mortality")
>>> current_data.load_exp_data()
>>> model = LSTM(expmodel_id=expmodel_id, n_batchsize=20, use_gpu=True,
    n_epoch=100) # initialize LSTM model and set hyperparameters
>>> model.fit(current_data.train, current_data.valid)
>>> model.inference(current_data.test)  #  inference on test data
>>> prediction_results = model.get_results() # retrieve the test results
>>> evaluation = func(prediction_results["hat_y"], prediction_results["y"])
```

Code Snippet 1: Demo of PyHealth API with Mortality Prediction

Figure 1: PyHealth contains three modules to support various healthcare Predictive tasks



Within the framework, a set of helper and utility functions (`check_parameter`, `label_check`, and `partition_estimators`) are included in the library for quick data and model exploration. For instance, `label_check` can automatically check the data label and infer the task type, e.g., binary classification and multi-classification.

3. Project Focus

Build robustness. We leverage continuous integration (e.g., *Travis CI*, and *CircleCI*) to conduct automated testing under various operating systems and Python versions.
Community-based development and project relevance. PyHealth 's code repository is hosted on GitHub[1] to facilitate collaboration. By the submission time, more than five people have contributed to the development and bug fix. PyHealth has been used in various commercial projects at IQVIA, a leading healthcare research institute; researchers at the University of Illinois Urbana-Champaign and Carnegie Mellon University are actively using the library for algorithm design and benchmark.
Documentation and examples. Comprehensive documentation is developed using `sphinx` and `numpydoc` and rendered using *Read the Docs*[2]. It includes detailed API references, an installation guide, code examples, and algorithm benchmarks.

4. Conclusion and Future Plans

This paper presents PyHealth , a comprehensive toolbox built-in Python for healthcare AI, with more than 30 classical and emerging learning models. A few future directions have been designed. First, we will enhance the library with specialized models for multimodal data. Second, the support of more different types of standard healthcare data formats will be provided. Additionally, we also plan to use PyHealth to generate a large-scale benchmark for healthcare predictive tasks.

1. `https://github.com/yzhao062/PyHealth`
2. `https://pyhealth.readthedocs.io/`

References

Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 65–74, 2017.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pages 3504–3512, 2016.

Junyi Gao, Cao Xiao, Lucas M Glass, and Jimeng Sun. Dr. Agent: Clinical predictive model via mimicked second opinions. *Journal of the American Medical Informatics Association*, 27(7):1084–1091, 06 2020a. ISSN 1527-974X. doi: 10.1093/jamia/ocaa074.

Junyi Gao, Cao Xiao, Yasha Wang, Wen Tang, Lucas M Glass, and Jimeng Sun. Stagenet: Stage-aware neural networks for health risk prediction. In *Proceedings of The Web Conference 2020*, pages 530–540, 2020b.

Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1):1–18, 2019.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Shenda Hong, Cao Xiao, Tengfei Ma, Hongyan Li, and Jimeng Sun. MINA: multilevel knowledge-guided attention for modeling electrocardiography signals. In *IJCAI*, pages 5888–5894, 2019.

Andrew Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. 04 2017.

G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. doi: 10.1109/CVPR.2017.243.

Shaoxiong Ji, Erik Cambria, and Pekka Marttinen. Dilated convolutional attention network for medical code assignment from clinical text. In *3rd Clinical Natural Language Processing Workshop at EMNLP*, 2020.

S. Lawrence, C. L. Giles, Ah Chung Tsoi, and A. D. Back. Face recognition: a convolutional neural-network approach. *IEEE Transactions on Neural Networks*, 8(1):98–113, 1997. doi: 10.1109/72.554195.

Fei Li and Hong Yu. Icd coding from clinical text using multi-filter residual convolutional neural network. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Zhi Li, Dengshi Zhou, Li Wan, Jian Li, and Wenfeng Mou. Heartbeat classification using deep residual convolutional neural network from 2-lead electrocardiogram. *Journal of Electrocardiology*, 58, 11 2019. doi: 10.1016/j.jelectrocard.2019.11.046.

Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1903–1911, 2017.

Taylor Miller, Michael Mastanduno, Levi Thatcher, and Taylor Larsen. healthcareai-py v1.0, August 2017. URL `https://doi.org/10.5281/zenodo.999013`. https://github.com/HealthCatalyst/healthcareai- py/releases/tag/v1.0.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. Explainable prediction of medical codes from clinical text. pages 1101–1111. Association for Computational Linguistics, June 2018. doi: 10.18653/v1/N18-1100.

Saman Parvaneh, Jonathan Rubin, Asif Rahman, Bryan Conroy, and Saeed Babaeizadeh. Analyzing single-lead short ECG recordings using dense convolutional neural networks and feature-based post-processing to detect atrial fibrillation. volume 39, page 084003, aug 2018.

Zhi Qiao, Austin Bae, Lucas M Glass, Cao Xiao, and Jimeng Sun. FLANNEL: Focal Loss Based Neural Network Ensemble for COVID-19 Detection. *Journal of the American Medical Informatics Association*, 10 2020. ISSN 1527-974X. doi: 10.1093/jamia/ocaa280. URL `https://doi.org/10.1093/jamia/ocaa280`. ocaa280.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and ZB Wojna. Rethinking the inception architecture for computer vision. 06 2016. doi: 10.1109/CVPR.2016.308.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. A label attention model for icd coding from clinical text. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3335–3341, 7 2020. doi: 10.24963/ijcai.2020/461.

E. K. Wang, X. Zhang, and L. Pan. Automatic classification of cad ecg signals with sdae and bidirectional long short-term network. *IEEE Access*, 7:182873–182880, 2019. doi: 10.1109/ACCESS.2019.2936525.

Cao Xiao, Edward Choi, and Jimeng Sun. Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review. *Journal of the American Medical Informatics*, 2018.

Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *CoRR*, abs/1611.05431, 2016. URL `http://arxiv.org/abs/1611.05431`.

Yanbo Xu, Siddharth Biswal, Shriprasad R Deshpande, Kevin O Maher, and Jimeng Sun. Raim: Recurrent attentive and intensive model of multimodal patient monitoring data. In *Proceedings of the 24th ACM SIGKDD international conference on Knowledge Discovery & Data Mining*, pages 2565–2573, 2018.

Yuxi Zhou, Shenda Hong, Junyuan Shang, Meng Wu, Qingyun Wang, Hongyan Li, and Junqing Xie. K-margin-based residual-convolution-recurrent neural network for atrial fibrillation detection. pages 6057–6063, 08 2019. doi: 10.24963/ijcai.2019/839.

Özal Yildirim. A novel wavelet sequence based on deep bidirectional lstm network model for ecg signal classification. *Computers in Biology and Medicine*, 96:189 – 202, 2018. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2018.03.016.