RESEARCH LETTER
Volume 4 (2023), No. 2, pp. 42-48
DOI: 10.15367/ch.v4i2.625

AnnoTS: An Annotation Tool for Sensor-Acquired Movement Data

COLE HAGEN, TESSA C. JOHNSON, SHIVAYOGI V. HIREMATH
Department of Health and Rehabilitation Sciences, Temple University, Philadelphia, Pennsylvania

Correspondence: cole.hagen0001@temple.edu (Cole Hagen)

Objective: Developing machine learning and deep learning models to detect aspects of human movement activity in naturalistic environments requires labeled datasets. AnnoTS aims to provide researchers with a software tool to annotate human movement data collected from wearable inertial measurement unit sensors. Methods: AnnoTS is a graphical user interface-based data annotation software created with Python libraries (PyQT5, PyQtGraph, and Pandas). Conclusion: AnnoTS facilitates the annotation of sensor-acquired movement data and is available as a standalone research software through an open-source code repository.

***Keywords**: Annotation, User interface, Human activity recognition, Machine learning*

Introduction

Advances in the field of wearable computing enable long-term monitoring of human movement in naturalistic environments. Accelerometers are the most common wearable sensors used to quantify human movement in clinical research and trials.[1,2] Accelerometer data is typically used to measure movement intensity and energy expenditure.[3] Although accelerometry-derived measures provide useful information regarding overall physical activity, they lack the ability to discriminate between different types of clinically relevant activities and movements.[4] Conversely, the combination of several inertial measurement unit (IMU) sensors, such as accelerometers combined with gyroscopes and magnetometers, can better inform the prediction of human activity.[5-7] Recent developments in machine learning and deep learning permit continuous monitoring of clinically relevant human movement activities over long periods of time.[5] However, continuous long-term monitoring generates large unlabeled time-series datasets that require appropriate data management techniques, including processes for annotating clinically important aspects of movement data.

Manual annotation of IMU data is necessary to develop supervised machine learning and deep learning models that can accurately predict human activity in naturalistic environments (see Fig. 1). Supervised machine learning and deep learning models require labeled data from the annotation process to teach models the correct labels to predict. Annotation is the process of attributing labels to events or sequences of raw data (i.e., data collected at large sampling rates such as 50 times a second or 50Hz) to provide context or meaning to the data.[8] Available data annotation software includes tools to label clinically relevant movement activities from, for example, video data capturing human behavior and gait patterns,[9] or physiological signals monitoring heart function (e.g., heart rate).[10] Two annotation tools are available to label IMU data: the Wearables Development Toolkit (WDK)[11] and Signaligner-Pro.[12] Both tools afford visualization, interaction, and manual segmentation of IMU sensor data; however, the WDK[11] requires users to understand and purchase MATLAB (i.e., programming software that requires a paid subscription), which may limit its usability. On the other hand, Signaligner-Pro[12] limits users to analyzing raw accelerometer data from Actigraph watches, thus excluding other types of sensor data such as gyroscope data and devices. Despite the wide availability of annotation software, there is a demand for open access (e.g., free software) and user-friendly annotation tools that can be used by researchers and clinicians who collect IMU data from multiple sensors (e.g., x-axis, y-axis, and z-axis accelerometer and gyroscope). Thus, we aim to address the limitations of previous IMU annotation software by creating Annotation Tool for Time Series Data (AnnoTS), which is a free standalone graphical user interface (GUI) software with modifiable code to annotate IMU data from multiple sensors. AnnoTS has the potential to allow researchers and clinicians to build robust labeled datasets, which are required to develop machine learning and deep learning applications.

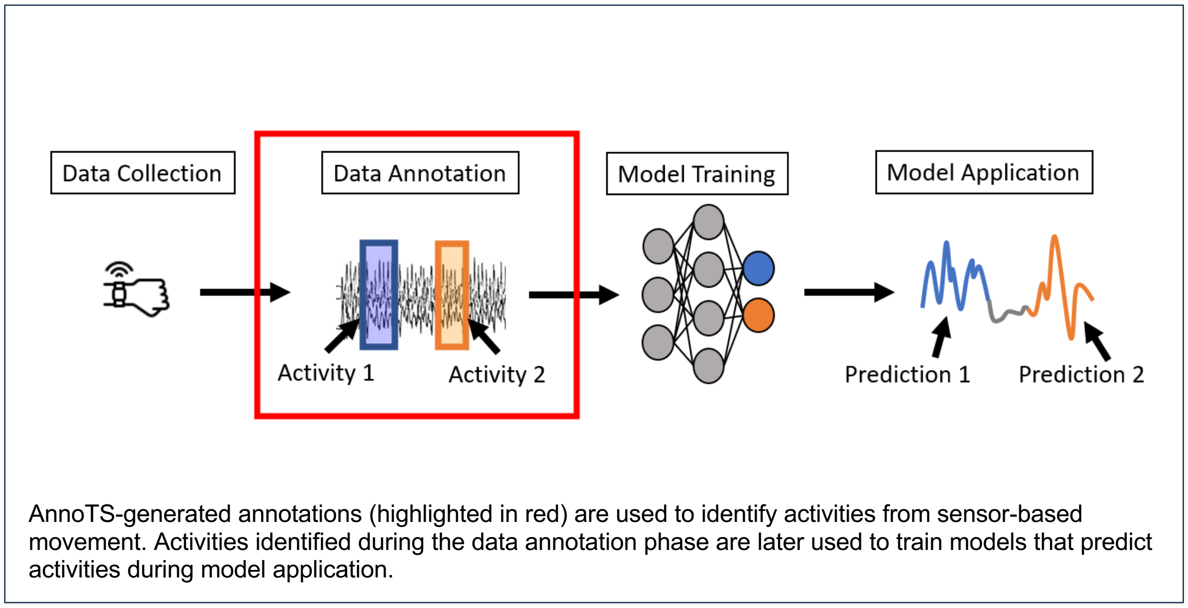

Data Collection
Data Annotation
Model Training
Model Application
Activity 1
Activity 2
Prediction 1
Prediction 2
AnnoTS-generated annotations (highlighted in red) are used to identify activities from sensor-based movement. Activities identified during the data annotation phase are later used to train models that predict activities during model application.

Figure 1: Example of human activity machine learning and deep learning model training and application procedures.

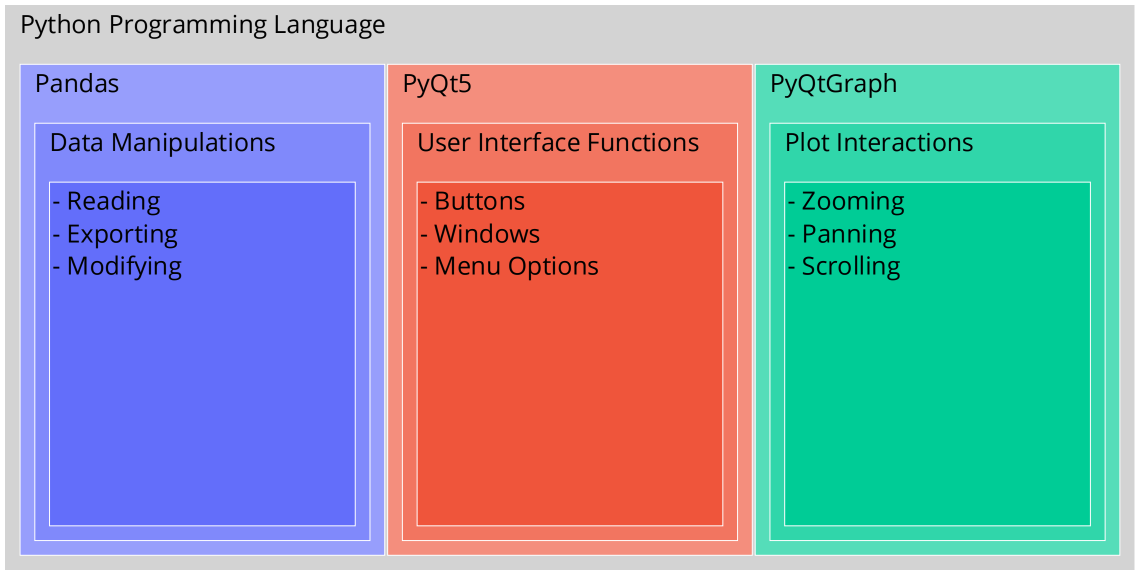
Materials and Methods

Software requirements

Data annotation software must satisfy various user requirements. First, the tool should operate with similar functionality for different data sources (e.g., accelerometer, gyroscope).[10] Also, the tool should support the loading of a variety of unique datasets with multiple IMU sensors. Further, the tool should take the form of a GUI to promote user accessibility, facilitate annotations in a short period of time, and reduce potential errors through the visualization of data.[10] Finally, the tool should allow users to seamlessly navigate their dataset (using multiple features such as zooming and scrolling) and visualize labeled data overlying time-series signals.[10]

Software architecture

All software architecture was conceptualized and designed by researchers in the Personal Health Informatics and Rehabilitation Engineering Lab at Temple University. Python programming language (Version 3.8)[13] and three Python libraries were implemented to enable specific functions within the GUI (see Fig. 2), including PyQT5, Pandas, and PyQtGraph. PyQT5 includes functions to create desktop applications. Pandas has functions to manipulate data structures and efficiently store and analyze data. PyQTGraph includes functions to visualize high-quality graphics on PyQt5 desktop applications with tools to create interactive plots. PyQT5, was used to enable user interaction, including all buttons, windows, and menu options within the AnnoTS GUI.[14] Second, Pandas was used to implement functions to read comma separated values (CSV) files, manipulate columns and rows of data, and export CSV files, including files of annotated and original data within AnnoTS.[15] Finally, PyQtGraph was used to generate a plot canvas that permits user interactions with the data plots and annotations (www.pyqtgraph.org) within AnnoTS.

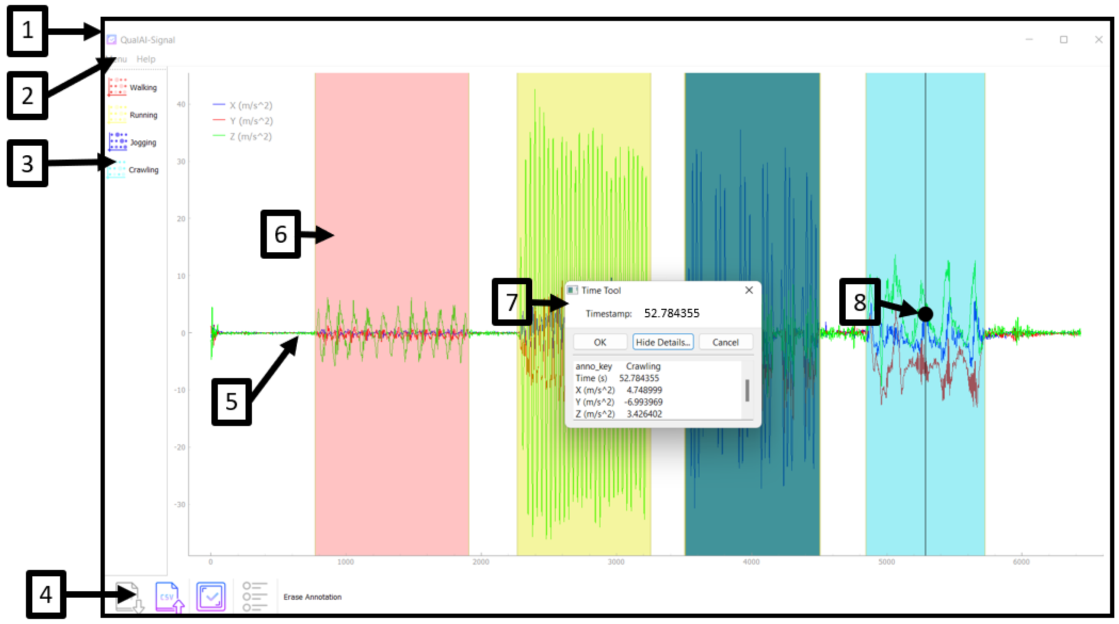


All software was built with the python programming language (highlighted in gray). PyQt5 (highlighted in blue) was responsible for user interactions. Pandas (highlighted in red) was responsible for data manipulation. PyQtGraph (highlighted in green) was used for plot interactions.

Figure 2: Software architecture and user interface functions

User interface

The user interface displays a large plot canvas to visualize IMU data and annotations (see Fig. 3). Signals from IMU sensor data, including a 3-axis accelerometer and 3-axis gyroscope, may be selected to display on the canvas. After importing IMU signals, the plot canvas allows multiple interactions, such as zooming and scrolling, manipulating annotation windows (creating, adjusting, and deleting), and exporting the resultant annotation data. The buttons at the bottom pane of the GUI contain options for importing files, configuring the number of annotations, confirming annotation window locations, deleting annotation locations, identifying indexed data (e.g., timestamps, specific accelerometer values, annotation labels), and exporting annotation data. The left pane of the GUI populates buttons corresponding to the number of annotations the user selects. For example, in the case of three annotations, three buttons will appear on the left pane of the GUI. Each button allocates an original label that symbolizes its encoding in the dataset (e.g., class one, class two, class three). Annotation labels may be modified to align with the type of activity or movement being annotated (e.g., running, pushing).



```
Timestamp: 52.784355

anno_key    Crawling
Time (s)    52.784355
X (m/s^2)   4.748999
Y (m/s^2)   -6.993969
Z (m/s^2)   3.426402
```

Main components of user interface: 1) Main window containing all user interface tools and windows. 2) Menu options for importing previously saved annotation files, saving progress on current annotation files, help, and launching a web browser for learning more about AnnoTS. 3) Customizable annotation labels used to add annotation windows to the plot canvas. 4) Buttons for importing CSV files, exporting CSV files, confirming annotation windows, erasing annotation windows, and adding new annotation labels. 5) Line plot of all imported IMU sensor data. 6) Annotation window color coded to correct customized label. 7) Window displaying raw IMU sensor data at a specified time stamp. 8) Slidable time tool to indicate time stamp for raw IMU sensor data viewing.

Figure 3: Components of the user interface

Use-Case

Prior work from our team used AnnoTS-generated annotations to train and test machine learning and deep learning models to classify sensor-acquired upper extremity activity (e.g., fine motor versus gross motor actions) from neurotypical individuals (see Fig 3.).[16] In the scope of this exploratory project, AnnoTS handled CSV files that ranged in size from 300,000-600,000 rows and seven columns, which consisted of one to two hours of IMU data collection. The data files, which contained six degrees of freedom, including three-axis acceleration and three-axes of angular velocity data from x, y, and z planes, along with timestamps from an ActiGraph GT9X Link watch, were imported into AnnoTS. For this project, our team was interested in accelerometer data and selected accelerometer signals in the x, y, and z planes to display on the primary plot canvas. Following this, we designated six task-based annotation categories (e.g., reaching, lifting, pushing and pulling, desktop activities, finger actions, and isometric actions); however, AnnoTS users may designate up to 17 annotation categories and name them to represent specific movement task events of interest (e.g., reaching, grasping). The primary plot canvas, which affords zooming and scrolling, allowed us to annotate the exact onset and offset times of task-based actions. The data were then exported to a CSV file format. While AnnoTS does not possess the capability to train machine learning and deep learning models, the annotation files were later used to train models to detect the different movement tasks.

Discussion and Conclusion

AnnoTS facilitates the manual annotation of large datasets with IMU sensors via a user-friendly GUI and is freely available as a standalone software application or code (https://github.com/chags1313/AnnoTS). While other annotation software exist, AnnoTS differs from previous IMU labeling software by offering open-source Python code and a free GUI that can handle multiple IMU signals. Users of AnnoTS can assign custom labels and implement unlimited annotations to segment any duration of IMU data on a large graphical display. Users can implement and edit annotations with manual input (i.e., time index of sensor data) or a modifiable annotation window. Moveable widgets further allow users to customize the GUI display. All data can be saved to a CSV format with annotations corresponding to timestamps of IMU data. The main graphical display provides several export options (e.g., .png, .jpg, .sav) for images of signal data with annotation windows.

AnnoTS has limitations to consider. First, AnnoTS has not been tested with datasets larger than 600,000 rows and more than seven columns of IMU sensor data. These data consisted of one to two hours of IMU data collection, which may vary widely among different research paradigms. Researchers should consider testing larger datasets in the future to identify specific limitations related to computational resources (e.g., operating system, random access memory, etc.). Second, AnnoTS limits users to CSV file formats for importing and exporting files. Depending on the IMU device used for data collection, file formats may include tabular separated values, excel files, text files, and others. Future work with AnnoTS should identify code modifications to accommodate for multiple file formats. Third, AnnoTS offers a maximum of 17 total label options. For research requiring more label options, it is possible to modify the code to accommodate for the specific number of labels needed. However, future work with AnnoTS should adjust the software architecture to enable more labeling options.

AnnoTS is a GUI-based data annotation software incorporating Python libraries (PyQT5, PyQtGraph, and Pandas) to label IMU sensor datasets for the subsequent prediction of human activity. AnnoTS is easy-to-use standalone open-source research software and modifiable code. Researchers and clinicians may benefit from using AnnoTS in their data pipeline to label data that is required for subsequent development and evaluation of new digital biomarkers in various populations of interest.

Conflicts of Interest
There are no conflicts of interest to disclose.

Funding Sources
This work was supported by The Assistant Secretary of Defense for Health Affairs endorsed by the Department of Defense through the Spinal Cord Injury Research Program under Award No. W81XWH-21-1-0637. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the Department of Defense.

Statement of Contributions
Mr. Hagen designed the software and wrote the paper. Ms. Johnson tested and contributed to the software and paper. Dr. Hiremath conceptualized the primary study, reviewed the software, and contributed to the paper.

References

1. Cervantes CM, Porretta DL. Physical activity measurement among individuals with disabilities: a literature review. Adapt Phys Act Q. 2010;27:173–90.
2. Ainsworth BE. How do I measure physical activity in my patients? Questionnaires and objective methods. Br J Sports Med. 2009;43:6–9.
3. Hey S, Anastasopoulou P, von Haaren B. Erfassung körperlicher Aktivität mittels Akzelerometrie—Möglichkeiten und Grenzen aus technischer Sicht. Bewegungstherapie Gesundheitssport. 2014;30:73–8.
4. Bonomi AG, Goris AH, Yin B, Westerterp KR. Detection of type, duration, and intensity of physical activity using an accelerometer. Med Sci Sports Exerc. 2009;41(9):1770-1777. doi:10.1249/MSS.0b013e3181a24536
5. Garofalo P. Healthcare applications based on MEMS technology. Adv Microelectron. 2012;39:24–8.
6. Dobkin BH. Wearable motion sensors to continuously measure real-world physical activities. Curr Opin Neurol. 2013;26:602–8.
7. Lowe SA, Ólaighin G. Monitoring human health behaviour in one's living environment: a technological review. Med Eng Phys. 2014;36:147–68.
8. Monarch, Robert Munro. Human-in-the-Loop Machine Learning: Active learning and annotation for human-centered AI. Simon and Schuster, 2021.
9. C. F. Martindale, N. Roth, J. Hannink, S. Sprager and B. M. Eskofier, "Smart Annotation Tool for Multi-sensor Gait-based Daily Activity Data," 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Athens, Greece, 2018, pp. 549-554, doi: 10.1109/PERCOMW.2018.8480193.
10. Fedjajevs A, Groenendaal W, Agell C, Hermeling E. Platform for Analysis and Labeling of Medical Time Series. *Sensors (Basel).* 2020;20(24):7302. Published 2020 Dec 19. doi:10.3390/s20247302
11. Haladjian J. The Wearables Development Toolkit: An Integrated Development Environment for Activity Recognition Applications. Proc. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4, Article 134 26 pages. 2019 Dec. doi:10.1145/3369813
12. Ponnada, A.; Cooper, S.; Tang, Q.; Thapa-Chhetry, B.; Miller, J.A.; John, D.; Intille, S. Signaligner Pro: A Tool to Explore and Annotate Multi-day Raw Accelerometer Data. In Proceedings of the 2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), Kassel, Germany, 22–26 March 2021; IEEE: Kassel, Germany, 2021; pp. 475–480.
13. Van Rossum, G., & Drake, F. L. Python 3 Reference Manual. Scotts Valley, CA: CreateSpace. 2009.
14. Summerfield, Mark. Rapid GUI Programming with Python and Qt: the Definitive Guide to PyQt Programming. Upper Saddle River, NJ: Prentice Hall, 2008.

15. McKinney, W., & others. Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56), 2010.
16. Johnson, T., & Hagen, C., Coffman, D.L., Merino, R., & Hiremath, S.V.Applying Machine Learning Algorithms to Classify Upper Extremity Actions Using Wrist-Worn Sensors: A Proof-of-Concept Study. Abstract accepted and poster session presented by Johnson at Temple University's College of Public Health Evidence-Based Practice Day, Philadelphia, Pennsylvania. 2022.