Automating the Development of Stress Detection Systems

Felix Beierle*†, Rüdiger Pryss†,
*National Institute of Informatics, Tokyo, Japan
†Institute of Clinical Epidemiology and Biometry, University of Würzburg, Würzburg, Germany
Email: felix.beierle@uni-wuerzburg.de, ruediger.pryss@uni-wuerzburg.de

***Abstract*—Stress is leading to bad health and contributes to economic loss due to employee absence. Real-time stress detection based on wearable sensor data can enable the implementation of mitigating strategies. While several approaches to stress detection exist, setting up a new system can be tedious. We demonstrate how the use of libraries and tools for automation can speed up many of the necessary steps when developing a stress detection system. We employ automated feature engineering and automated machine learning. The resulting stress detection system we developed this way is based on the WESAD dataset and achieves a F1 score of 0.87 for unseen users based on 30 seconds of wearable sensor data.**

***Index Terms*—stress detection, machine learning, AutoML, time series data, wearables, feature engineering**

Extended Abstract/Poster Research Paper

I. Introduction and Related Work

Stress contributes to bad health and bad work performance. It is estimated that 2–16% of a team's salaries are being spent on staff absence each year [1]. Automatically detecting stress could enable employers or individuals to implement mitigating strategies. Wearables are commonly used and contain a variety of sensors that could be used as data sources for stress detection, and, in general, users seem quite willing to share their data [2], [3]. In their extensive survey on stress detection, Can et al. showed that there are many machine learning models that researchers have used for detecting stress from sensor data [4]. The variety of available methods might be overwhelming when implementing a stress detection system for sensor data from wearables. However, there are also several specialized libraries and tools for automating processes needed for solving machine learning problems. In this paper, we demonstrate how a stress detection system including a deployed demonstrator can be set up with the help of automated library and tools, while achieving a good performance.

II. Methodology

A. Data Sets

We used two publicly available, labeled datasets in our work. Both of them are from studies that utilized the Empatica E4 wristband wearable. Specifically, we used data from the following sensors, recorded in the given frequencies:

This work was supported by a fellowship within the IFI program of the German Academic Exchange Service (DAAD). Corresponding author: Felix Beierle (felix.beierle@uni-wuerzburg.de).

- ACC – accelerometer sensor – 32Hz
- BVP – blood volume pulse – 64 Hz
- EDA – electrodermal activity – 4 Hz
- TEMP – temperature – 4 Hz

The first dataset is the WESAD dataset, for which Schmidt et al. collected data in a lab setting [5]. The 15 participants were wearing a Empatica E4 wristband and another sensor on the chest. The researchers utilized the Trier Social Stress Test (TSST) [6] for stress recordings, a standard test known for inducing stress. For our purposes, we disregarded other recorded states like *amusement* and only used the *baseline* and *stress* recordings. As we want to pursue the detection of stress from common wearable sensors, we used only the data from the wristband, not the chest sensor.

In [7], Hosseini et al. equipped 15 nurses with Empatica E4 wristbands during the COVID-19 outbreak. The nurses could edit stress labels that were predicted by a model, or fill out a questionnaire for labeling time windows. Stress was recorded as *no stress*, *medium*, or *high*. In our work, we only used the time windows marked as *no stress* and *high stress*.

B. Feature Engineering

We employed FLIRT for automated feature engineering for time series data [8]. FLIRT was specifically developed for wearable data and specifically supports the Empatica E4. FLIRT can calculate several features on sliding windows on time series data. There are two parameters to consider: the *window size* and the *step size*. The *window size* indicates how many seconds of data are used for the calculation of features. The *step size* indicates how many seconds the window moves ahead for the next feature calculation.

The smaller the *window size*, the fewer data points are needed for making a prediction, and the faster we can possibly detect stress. On the other hand, the smaller the *window size*, the less accurate the model will be. Consider an extreme case: with an extremely small window of only one value, it is unlikely that the calculated features will be representative of a stressful or unstressful situation.

C. Machine Learning

Overall, we have a binary classification problem: *stress* vs. *no stress*. For the Nurses dataset, we have 84% *stress* data vs. 16% *no stress* data from 15 users. For the WESAD dataset, we have 36% *stress* data vs. 64% *no stress* data from 15 users.

Because of the imbalance, we chose the F1 score instead of accuracy as the evaluation metric.

Our goal is to have a generic, user-independent model, i.e., a model that is trained on some users, that generalizes well, and is applicable to new, unseen users. For the machine learning pipeline, the requirement then is to have each user either in the train *or* in the test set.

For finding an appropriate machine learning pipeline, we employed AutoML (automated machine learning) with the help of TPOT [9], [10]. For the final model, for model explainability, we employed SHAP [11], [12]. All code is publicly available on GitHub.[1] We deployed a demonstrator showcasing the resulting model. We used FastAPI, Streamlit, and Docker.

III. Results

Feature engineering yielded more than 200 features overall. We removed all features with a correlation higher than 0.8. We calculated features for a variety of *window sizes* and *step sizes*. We used 12 users for training and 3 users for testing.

Our experiments immediately showed that we cannot build a model that detects stress from the Nurses dataset. The stress detection based on the Nurses dataset was not better than chance (random guessing). We suspect that the labeling was too imprecise. When the nurses reflected after a task if it was stressful, maybe larger time windows were labeled as stressful, while only parts of those were where the actual stress was experienced. We proceed only with the WESAD dataset.

Running TPOT only on the WESAD data with different *window* and *step sizes*, different tree-based algorithms showed promising results, especially the ExtraTreesClassifier.[2] Trading off between less data needed for stress detection versus a lower F1 score, we chose a 30 second *window size* and 1 second *step size* for the final dataset.

Removing the correlated features, we have 71 features in the final dataset. The final pipeline contains two steps, a MaxAbsScaler[3] and an ExtraTreesClassifier. After a hyperparameter search with BayesSearchCV[4], we set *bootstrap* to True, *criterion* to *log_loss*, *max_features* to 0.6, *minsamples_leaf* to 5, *min_samples_split* to 14, and *n_estimators* to 50. The final scores on the unseen test users are a F1 score of 0.87, with a precision of 0.94 and a recall of 0.82.

We investigated feature importance with SHAP. We observe that the *permutation entropy* (measure of complexity of time series data [13]) of the BVP (blood volume pulse) signal is deemed the feature with the most significant impact on the model output. For lower values for this feature, the model is likelier to predict 1 (stress). Given the data we have, it is very difficult for a layperson of the medical field to interpret results like this. What we could imagine is that if there is a high stress level, the pulse is generally high, driving down the complexity of the BVP values. Similarly, we could guess the interpretation of the other features – a medical expert should be involved for more knowledgeable interpretations of these results. While it might be helpful or interesting to the user of a stress detection system to learn about the reasons for a particular prediction/detection, such results from SHAP are most likely difficult to interpret.

For demonstration purposes, we developed a web interface, allowing the user to interactively query the deployed model. We deployed the demonstrator via Docker Compose with three separate Docker containers. One handles processing of the sensor data, one contains the model for inference, and one contains the Streamlit web-interface.

IV. Discussion

Overall, utilizing specialized libraries can save a lot of time when developing stress detection systems. With FLIRT, we could successfully automate feature engineering for time series data from wearable sensors. TPOT proved to be a valuable tool for finding an appropriate machine learning pipeline. The trade-off between faster prediction and lower F1 score when choosing a *window size* depends on the type of data and specific type of application to be deployed. SHAP for complex time series features might not yield meaningful results for laypersons. Frameworks like FastAPI and Streamlit enable the quick development of demonstrators. Deploying the components separately allows the individual scaling of each service.

Based on our observations on the different performance of the two datasets, we suspect that precise labeling of the stressful situation is necessary. There are only 15 participants in the WESAD dataset. Future work includes collecting more data to get more robust results. Conducting lab sessions for doing this will likely be costly in terms of time and money. The WESAD study was conducted with the Empatica E4. We am not sure to what extent the results are applicable to other wristbands like the Apple Watch. Future work includes collecting data from multiple wristbands and trying to replicate the results. When collecting more data, we could collect data with labels indicating a level of stress instead of binary *stress*/*no stress* values. Then we could predict a stress level instead of just a binary label.

Overall, many steps of the data processing and machine learning pipeline can be automated, enabling researchers and developers to quickly develop first solutions for stress detection systems or for similar scenarios.

[1] https://github.com/fbeierle/stress-detection

[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html; accessed 2023-05-05

[3] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MaxAbsScaler.html; accessed 2023-05-05

[4] https://scikit-optimize.github.io/stable/modules/generated/skopt.BayesSearchCV.html; accessed 2023-05-05

References

[1] S. Bevan, *Costing Sickness Absence in the UK*. Institute for Employment Studies, 2001.

[2] F. Beierle, V. T. Tran, M. Allemand, P. Neff, W. Schlee, T. Probst, J. Zimmermann, and R. Pryss, "What data are smartphone users willing to share with researchers?" *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 2277–2289, 2020.

[3] F. Beierle, J. Allgaier, C. Stupp, T. Keil, W. Schlee, J. Schobel, C. Vogel, F. Haug, J. Haug, M. Holfelder, B. Langguth, J. Langguth, B. Riens, R. King, L. Mulansky, M. Schickler, M. Stach, P. Heuschmann, M. Wildner, H. Greger, M. Reichert, H. A. Kestler, and R. Pryss, "Self-Assessment of Having COVID-19 With the Corona Check mHealth App," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 6, pp. 2794–2805, Jun. 2023.

[4] Y. S. Can, B. Arnrich, and C. Ersoy, "Stress detection in daily life scenarios using smart phones and wearable sensors: A survey," *Journal of Biomedical Informatics*, vol. 92, p. 103139, Apr. 2019.

[5] P. Schmidt, A. Reiss, R. Duerichen, C. Marberger, and K. Van Laerhoven, "Introducing WESAD, a Multimodal Dataset for Wearable Stress and Affect Detection," in *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, ser. ICMI '18. Association for Computing Machinery, Oct. 2018, pp. 400–408.

[6] C. Kirschbaum, K. M. Pirke, and D. H. Hellhammer, "The 'Trier Social Stress Test'–a tool for investigating psychobiological stress responses in a laboratory setting," *Neuropsychobiology*, vol. 28, no. 1-2, pp. 76–81, 1993.

[7] S. Hosseini, R. Gottumukkala, S. Katragadda, R. T. Bhupatiraju, Z. Ashkar, C. W. Borst, and K. Cochran, "A multimodal sensor dataset for continuous stress detection of nurses in a hospital," *Scientific Data*, vol. 9, no. 1, p. 255, Jun. 2022.

[8] S. Föll, M. Maritsch, F. Spinola, V. Mishra, F. Barata, T. Kowatsch, E. Fleisch, and F. Wortmann, "FLIRT: A feature generation toolkit for wearable data," *Computer Methods and Programs in Biomedicine*, vol. 212, p. 106461, Nov. 2021.

[9] R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore, "Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, ser. GECCO '16. Association for Computing Machinery, Jul. 2016, pp. 485–492.

[10] T. T. Le, W. Fu, and J. H. Moore, "Scaling tree-based automated machine learning to biomedical big data with a feature set selector," *Bioinformatics*, vol. 36, no. 1, pp. 250–256, Jan. 2020.

[11] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[12] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee, "From local explanations to global understanding with explainable AI for trees," *Nature Machine Intelligence*, vol. 2, no. 1, pp. 56–67, Jan. 2020.

[13] C. Bandt and B. Pompe, "Permutation Entropy: A Natural Complexity Measure for Time Series," *Physical Review Letters*, vol. 88, no. 17, p. 174102, Apr. 2002.