Adaptive Pooling Is All You Need: An Empirical Study on Hyperparameter-insensitive Human Action Recognition Using Wearable Sensors

Mubarak G. Abdu-Aguye
Department of Computer Engineering
Ahmadu Bello University
Zaria, Nigeria
mubarak.abduaguye@ejust.edu.eg

Walid Gomaa
Computer Science and Engineering
Egypt-Japan University of Science and Technology
New Borg el-Arab, Egypt
walid.gomaa@ejust.edu.eg

Yasushi Makihara
ISIR
Osaka University
Osaka, Japan
makihara@am.sanken.osaka-u.ac.jp

Yasushi Yagi
ISIR
Osaka University
Osaka, Japan
yagi@am.sanken.osaka-u.ac.jp

***Abstract*—A plethora of techniques have been proposed in human action recognition fields, and particularly deep learning-based methods such as convolutional neural networks (CNNs) have achieved impressive results. Usually, there is need to tune hyper-parameters in the deep neural network (e.g., filter size, stride) to achieve reasonable results. Such hyper-parameter tuning is, however, extremely time and resource-intensive even for small models. In this paper, we posit that the inclusion of an adaptive pooling in CNNs used for human action recognition largely eliminates the need for hyper-parameter tuning. Specifically, we demonstrated our idea for human action recognition using inertial sensor data (i.e., a temporal sequence) with a one-dimensional adaptive pooling. We compared the adaptive pooling to conventional CNNs with randomly chosen hyper-parameters using a publicly available data set for human action recognition. Experimental results showed that the adaptive pooling achieved better accuracy than the conventional CNNs.**

***Index Terms*—activity recognition, convolutional neural networks, hyperparameters, adaptive pooling**

I. Introduction

Human action recognition is a challenging task. Traditional approaches to solving the problem include the use of video data capturing the visual patterns of different actions. More recently however, Inertial Measurement Unit (IMU) sensor data has been adopted for this same purpose with impressive results [1], [2]. This also comes with the advantages of mobility and improved user privacy. As such, we direct the proceeding discussion to IMU sensor-based activity recognition.

Regardless of the modality adopted, Human action recognition is beset by different concerns including feature extraction and classification modalities. Traditionally, Feature extraction for IMU-based sensor data was handled using different statistical and temporal measures, including but not limited to entropy, skewness, kurtosis, etc., with the selection of features usually determined empirically. Although good performance was obtained, the problem of feature extraction and selection remained pertinent.

With the advent of deep neural networks, specifically convolutional neural networks (CNNs), the need for manual feature extraction is obviated. This is due to the ability of the convolutional portions of such networks to automatically learn the optimal features which aid the downstream task (i.e., classification, regression, etc.). This has led to massive performance improvements in several domains, such as: WiFi localization [3], Radio communications [4] and Video Translation [5].

The prevalence of deep neural networks motivates the question of how to obtain the best performance (e.g. predictive accuracy, etc.) from their use. This could be seen to consist of two distinct components. The first of these is the problem of finding the "best" architecture for the given problem, which is formally termed Neural Architecture Search (NAS) [6]. The second component is concerned with finding the optimal configuration for a given network architecture. This configuration refers to the *hyperparameters* of the network, which determine the structure of the network. For instance, the number of recurrent units to use in Recurrent Neural Networks (RNNs) or the number, size, and stride of convolutional filters in Convolutional Neural Networks (CNNs). However, network architectures and hyperparameters need to be tuned for individual problems, and cannot, in general, be inferred automatically in a theoretically-principled manner.

In the case of NAS, unless particular constraints are placed on the network architecture, there are virtually infinite pos-

Walid Gomaa is supported by the Information Technology Industry Development Agency (ITIDA) under the ITAC Program Grant no. PRP2019.R26.1 - A Robust Wearable Activity Recognition System based on IMU Signals.'

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

sible architectures that may yield optimal performance on the given problem. Additionally, it is immediately clear that it is computationally-expensive due to the need to train the candidate models from scratch each time. Additionally, NAS is, to some extent, still subject to the hyperparameter search problem i.e. regardless of the architecture chosen, proper hyperparameters must be selected to yield good performance on the problem. This also implies that there is no theoretical guarantee that a simpler architecture with proper hyperparameters cannot match or outperform the optimal architecture as determined by a NAS procedure.

We believe that the preceding discussion sets the stage for emphasizing the relative benefit of focusing on hyperparameter search for deep networks. Given the representational capacity of deep networks in general, it is not unreasonable to instead attempt to tune them for optimal performance rather than focus solely on their particular architecture. This is because hyperparameter tuning scales with the complexity of the network, and therefore its difficulty can be somewhat contained by simplifying the network architecture as much as possible. Although the hyperparameter search problem is, like NAS, basically infinite, it is much more tractable in practice because it inherently requires discrete and not continuous solutions, and the search space can be limited to regions which contain a (small) range of values. Therefore, we believe that considering a fairly simple deep network and tuning that accordingly is a much more feasible approach to obtaining good performance on a range of different tasks.

There are existing techniques designed to mitigate the hyperparameter search problem. From first principles, Grid-search [7] is one such approach that may be applied to the hyperparameter optimization problem, which considers a subregion of the (discrete) hyperparameter space, and visits every point in this region with a view to finding the optimal hyperparameter configuration therein. This is something of a naive approach, necessitating the use of less computationally intensive approaches and methods. Random search [8] is another approach, which involves the visitation of random points within some region of the hyperparameter space. This approach, being non-exhaustive, may not yield the optimal results but rather 'good-enough' results depending on the criteria set. More advanced techniques like Bayesian hyperparameter optimization [9] also exist but are not applicable to all problems, in that they rely on certain assumptions that cannot easily be guaranteed to be universally valid. Regardless, the previously-described techniques still remain resource-intensive due to the need to train and evaluate multiple models, although countermeasures like parallelization may be adopted to mitigate this. In the presence of these drawbacks, two questions evolve:

- Is it possible to derive a technique that yields optimal model hyperparameters without the associated computational overhead?
- Is it possible to design models that are robust to hyperparameter settings, such that the exact selection of hyperparameters does not affect the performance of the model significantly?

Therefore, the work done in this paper is aimed towards providing an answer to the second question. We propose the inclusion of *adaptive pooling* in convolutional neural networks applied to activity recognition based on IMU-based sensor data. Adaptive Pooling is a technique to perform multi-scale summarization over convolutional feature maps, while capturing the essential behavior of the feature map itself. It has the effect of reducing the amount of data needed in tuning the downstream portion of CNNs, while reducing the convergence time and increasing generalization performance. To the best of our knowledge, this is also the seminal work in which adaptive pooling is adopted for the realization of hyperparameter-robust deep convolutional networks. We design a suitable experimental methodology to validate our proposition, and carry out experiments on a publicly-available dataset to this end. Empirical results obtained from our experiments indicate the efficacy of the proposed method in realizing the stated goal, consistently providing near-optimal model performance in the worst case.

The rest of this paper is organized as follows. Section II provides some background on concepts relevant to the work done in this paper. Section III introduces our experimental methodology and its particulars. In Section IV, we present the results obtained and interpret them accordingly. We conclude the paper and describe points for future consideration in Section V.

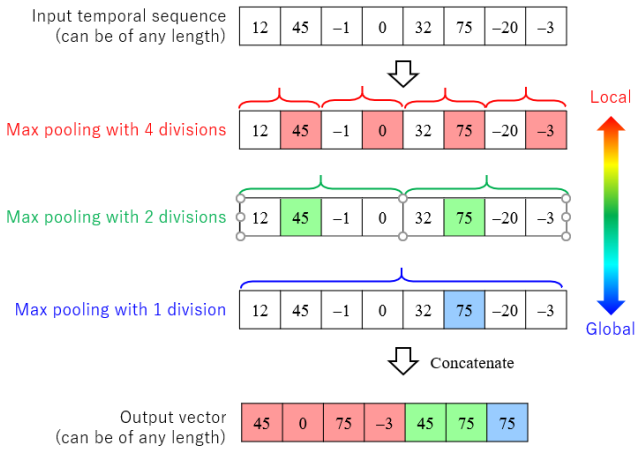

Fig. 1. Spatial pyramid pooling for 1D data (4-2-1 pooling sizes)

II. Background

A. Adaptive Pooling

Adaptive pooling is a generalization of another technique called spatial pyramid pooling, which was first introduced in [10]. Spatial pyramid pooling was introduced to solve the problem of varying input sizes in CNNs for image-based tasks, and therefore involves the conversion of convolutional feature maps of varying sizes into fixed-length summarizations. These fixed-length summarizations are compatible with the fully-connected portions of such networks which are actually the

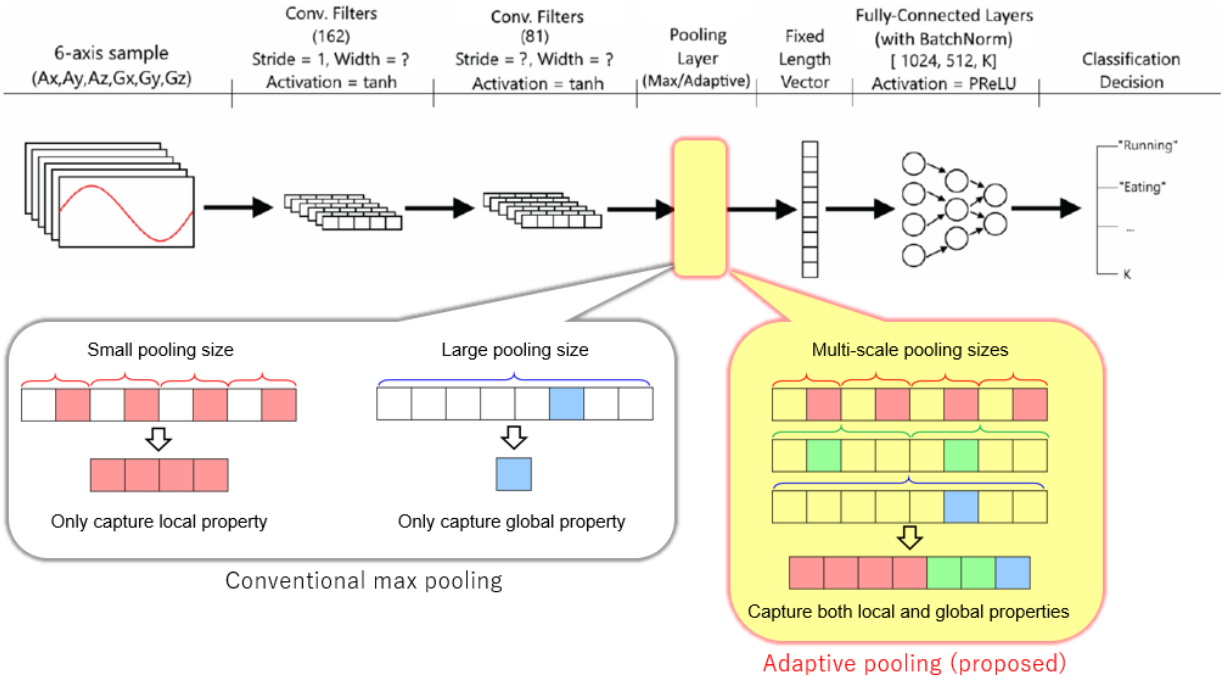

Fig. 2. System Architecture.

network portions enforcing the requirement for fixed-length inputs.

The summarization itself is performed over multiple scales. That is, the feature map is broken into a number of fixed-size regions, over which a simple pooling operation (max or average pooling) is carried out, thereby yielding a fixed number of features. This is typically done over multiple varying region sizes, ranging from global (i.e., over the whole image), to smaller sizes. At each size, the number of features derived is different, but captures details at different levels of granularity about the input data. By concatenating the features obtained at different region sizes (i.e., scales), a compact yet informative descriptor is obtained.

Although the preceding discussion is focused on 2-dimensional feature maps, it can be seen that the same principle can be applied to multidimensional inputs as well. Therefore, adaptive pooling simply describes the extension of spatial pyramid pooling to arbitrary dimensions, although we focus on the one-dimensional case in this work due to the nature of the signals involved. In this case, each of the resulting 1-D feature maps is divided into regions, over which a pooling operation (i.e., max or average pooling) is performed, similar to the 2-D case. This is illustrated in Figure 1.

In this work we use a simple notation to describe the parameters/configuration of the adaptive pooling operation, which effectively determines the (number of) levels/region sizes at which adaptive pooling is performed. For simplicity, we use hyphens to denote the sizes used i.e., 4-2-1 implies that the input is broken into 4 regions, each of which is pooled, then 2 regions, then a global pooling (i.e., 1 region) is performed. In general for a configuration x-y-z, the number of features obtained from the use of adaptive pooling is $x + y + z$.

TABLE I
HYPERPARAMETER CONFIGURATIONS CONSIDERED

Hyperparameter Name	Values
Conv1 Size	2,3,4,5,6,7,8,9,10,11,12,13,14,15
Conv2 Size	2,3,4,5,6,7,8,9,10,11,12,13,14,15
Conv2 Stride	1,2,3,4,5,6,7,8
MaxPool Size	2,3,4,5,6,7,8
MaxPool Stride	1,2,3,4,5,6,7,8
Total Configurations:	87,808

III. EXPERIMENTAL SETUP

A. Dataset

For purposes of clarity, we perform our experiments on the REALDISP dataset ([11], [12]), which is a dataset consisting of 33 different activities collected from 17 individuals in total at a sample rate of 50Hz. REALDISP includes 4 sensor modalities (i.e., accelerometer, gyroscope, magnetometer and rotation), but we consider only the first two in this work. We preprocess the dataset into fixed-length windows, each 1 second long and label each window according to the activity it corresponds to. This yields a total number of 13,963 samples, 75% of which are used for training the models with the remainder used in testing the predictive performance of the models. It is also pertinent to state that, during each of the n training and testing steps mentioned previously, a different training and testing split is considered in order to obtain a better estimate of the model's true performance. We select $n = 5$ in this work.

B. Protocol

The core hypotheses governing this work are itemized as follows:

- Hypothesis 1: Given any arbitrary convolutional neural network architecture, including adaptive pooling in it gives comparable or better results than the vanilla architecture i.e., without adaptive pooling.
- Hypothesis 2: Considering some subregion of the hyperparameter space, the variance in performance of the corresponding models which include adaptive pooling is smaller than the corresponding models without it, due to the hyperparameter-agnosticism granted by adaptive pooling.
- Hypothesis 3: The performance of the models including adaptive pooling are very close to the best performance obtained by any of the considered models i.e., adaptive pooling situates the model performance at or very close to the performance optimum achieved by the best model amongst the configurations evaluated.

Therefore, in line with the above hypotheses, we design our experimental protocol as follows:

- We design a benchmark convolutional architecture (shown in Figure 2), consisting of a number of convolutional layers and a single max-pooling layer. Since we are not considering neural architecture search, we fix the base architecture and instead adjust the hyperparameters of the network layers. We adopt this architecture as it has been used successfully in solving the activity recognition problem with generally good results [13], [14], and is sufficiently small to feasibly demonstrate the concept under discussion.
- We define a subregion of the hyperparameter space which contains reasonable hyperparameter configurations for the problem at hand. The hyperparameters and the values considered for each are given in Table I.
- Given the large number of candidate configurations in this subregion (a total of 87,808), we randomly sample a number of configurations to reduce the computational resources required for the experiments while giving an unbiased estimate of the expected behavior of the proposed method. Results from 2,743 models (3.12% of the total) are obtained.
- For each candidate configuration c, we:
  - Initialize the benchmark architecture's layers with the elements of the configuration c, then obtain the mean value m_c of the performance metric (i.e., prediction accuracy) over a number of runs n, each consists of a training and a testing step.
  - Using the same configuration, we replace the default pooling operator - i.e. traditional max-pooling - in the architecture with an adaptive pooling layer with a fixed hyperparameter configuration (i.e., 4-2-1, using the notation described previously), then obtain the mean value m'_c of the performance metric using this modified version of the architecture, also over n training and testing steps.
- We then compare m_c and m'_c and aggregate the statistics over all considered c candidates. Based on the derived statistics, we then validate each of the foregoing hypothesis accordingly.

All the experiments were implemented using the PyTorch [15] neural network library, version 1.0.

We present the results obtained and their interpretation in the following section.

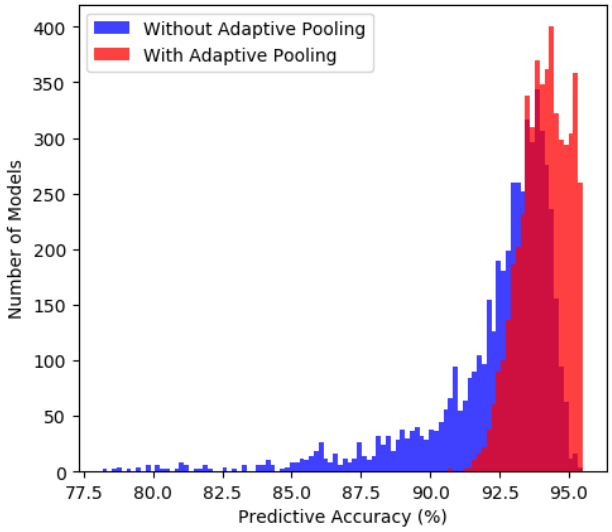

Fig. 3. Distribution of Model Performances With/Without Adaptive Pooling

IV. Results and Discussion

In order to streamline the proceeding discourse, we present the results obtained graphically. We derive a histogram over the model performances as shown in Figure 3, showing the statistical distribution thereof between the proposed and comparative (i.e., without adaptive pooling) models. Additionally, we include a scatter plot of the baseline model performances against the proposed models' performances, shown in Figure 4.

It can be observed that, when the proposed method is used, the mean of the models' performances is higher than the mean of the baseline model (94.14% vs. 92.31%). This is in line with the first hypothesis put forth, and may be attributed to the inherent benefits of the adaptive pooling - its ability to effectively summarize the input data, maintaining its key characteristics while eliminating much of the details which may be uninformative (or even misleading to the classifier). This directly improves generalization as it permits the classification portion of the network focus on the core parts of the input which are beneficial for the downstream classification, rather than factoring in a large number of features which may eventually degrade its performance based on their noisiness. Additionally, the summarized version of the input also reduces

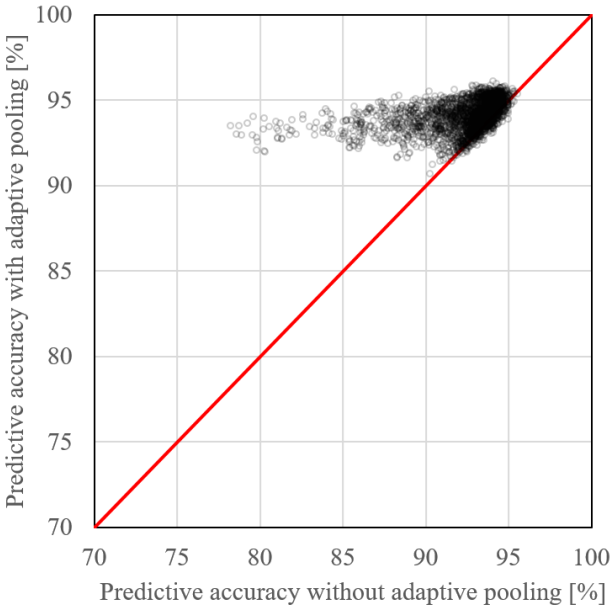

Fig. 4. Scatter plot of model performances with/without adaptive pooling. The Red diagonal line indicates a trade-off between accuracies with/without adaptive pooling, i.e., plots above the line means that accuracies with adaptive pooling are better than those without adaptive pooling.

the number of weights utilized in the classification portion of the network, allowing for faster convergence. Figure 4 also shows the majority of the points distributed above the red line, indicating that for a given configuration, the performance obtained from the proposed method is generally higher than that obtained using the baseline model.

Additionally, it is also apparent that the spread of the models' performances when using the proposed method is much smaller than that without adaptive pooling, which can also be seen from the spread of the points for the horizontal (i.e., without average pooling) and the vertical (i.e., with average pooling) directions, in Figure 4. This indicates that the variance of the models which include the proposed method ($0.778\%^2$) is much smaller than the models which do not include it ($6.09\%^2$). This is in line with the expectations put forth by the second hypothesis. This occurs because unlike the baseline models (i.e., which include max-pooling), the inclusion of adaptive pooling lends significant robustness against the hyperparameters chosen, allowing the models to maintain their performance even in the presence of disparate hyperparameter settings. This is by virtue of its ability to summarize the input data across multiple timescales and therefore preserve the salient information in a robust way.

Going further, it should be noted that the small variance in the performances obtained from the proposed models also indicates that the models are reasonably comparable to the very best model obtained, although a slight performance penalty may be observed. This validates the third hypothesis, since their actual performance values lie close (i.e., within 4-5%) to the maximum possible performance attainable (from using this architecture).

From the preceding discussion, it can be seen that each of the proposed hypotheses have been confirmed, indicating the validity of the initial proposition. Although the results presented have been aggregated over a fraction of the total hyperparameter space, we ensure that random sampling in some sense considers the entirety of the desired subregion and therefore provides a statistically-valid view of the performance of the proposed method.

V. Conclusion and Future Work

In this work, we have proposed an empirical adjustment for convolutional neural networks as applied to wearable sensor-based human activity recognition, which aims to eliminate the need for hyperparameter tuning. We design an experimental methodology to empirically validate our hypothesis, by considering the results obtained from a baseline CNN architecture using differently and randomly sampled hyperparameter configurations. Next, we compare them to the results obtained from a version of the same architecture modified based on the proposed method. We obtained results indicating that the proposed modification yields generally better and more stable/consistent results than the baseline architecture, thereby validating the preceding approach and establishing its suitability for the domain in question.

In the future we intend to perform much more comprehensive evaluations of different models and also investigate the potential of adaptive pooling in achieving the same or similar effects for data types at different dimensionalities e.g. images.

References

[1] B. Barshan and M. C. Yüksek, "Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units," *The Computer Journal*, vol. 57, no. 11, pp. 1649–1667, 2014.

[2] S. Ashry, R. Elbasiony, and W. Gomaa, "An lstm-based descriptor for human activities recognition using imu sensors," in *Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics, ICINCO*, vol. 1, 2018, pp. 494–501.

[3] H. Rizk, M. Torki, and M. Youssef, "Cellindeep: Robust and accurate cellular-based indoor localization via deep learning," *IEEE Sensors Journal*, vol. 19, no. 6, pp. 2305–2312, 2018.

[4] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep learning for massive mimo csi feedback," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 748–751, 2018.

[5] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Evaluating surgical skills from kinematic data using convolutional neural networks," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 214–221.

[6] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *Journal of Machine Learning Research*, vol. 20, no. 55, pp. 1–21, 2019.

[7] D. C. Montgomery, *Design and analysis of experiments*. John wiley & sons, 2017.

[8] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[9] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[11] O. Banos, M. Damas, H. Pomares, I. Rojas, M. A. Toth, and O. Amft, "A benchmark dataset to evaluate sensor displacement in activity recognition," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, ser. UbiComp '12. New York, NY, USA: ACM, 2012, pp. 1026–1035. [Online]. Available: http://doi.acm.org/10.1145/2370216.2370437

[12] O. Banos, M. A. Toth, M. Damas, H. Pomares, and I. Rojas, "Dealing with the effects of sensor displacement in wearable activity recognition," *Sensors*, vol. 14, no. 6, pp. 9995–10 023, 2014. [Online]. Available: http://www.mdpi.com/1424-8220/14/6/9995

[13] M. G. Abdu-Aguye and W. Gomaa, "Competitive feature extraction for activity recognition based on wavelet transforms and adaptive pooling," in *2019 International Joint Conference on Neural Networks (IJCNN)*, July 2019, pp. 1–8.

[14] M. G. Abdu-Aguye and W. Gomaa, "Versatl: versatile transfer learning for imu-based activity recognition using convolutional neural networks," in *The 16th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2019.

[15] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf