Characterizing Types of Convolution in CNN for Step Detection Using IMU data

LIPING YIN, University of Twente, The Netherlands

Step detection is essential to realizing a well-functioned pedometer-enabled application. Recent works of IMU-based step detection have shown that Convolutional Neural Network (CNN or ConvNet) is promising in dealing with sequential time series data and results in good step detection accuracy. However, no related works have been done to compare different convolution types used in the CNN model for step detection. This paper investigated and compared the performance (e.g., model accuracy and complexity) of various convolution types (e.g.,1D, 2D, 3D) in a standard shallow CNN model for step detection. We conducted the experiments using only accelerometer data under regular walking mode from a public IMU dataset that contains records of 30 participants under different walking scenarios. Results showed that the 1D convolution type has relatively higher accuracy than 2D and 3D convolutions, and under a 2D convolution context, the depthwise separable convolution type generated the lowest model complexity.

Additional Key Words and Phrases: Convolutional Neural Network (CNN), step detection, Inertial Measurement Unit (IMU), Time-serial data, Pedometer

1 INTRODUCTION

Modern pedometers (or step counters) can measure distance and count steps. There are many use cases of pedometers in daily life, such as health monitoring, fitness, navigation through Pedestrian Dead Reckoning (PDR) systems, and context awareness [7, 10]. For example, in indoor navigation, the state-the-art PDR navigation determines the user's current position by deriving the characteristics of human gait locomotion such as the number of steps, step length, and direction [12, 13].

Pedometers detect steps by measuring motion through the use of Inertial Measurement Units (IMUs), which are sensors (e.g., accelerometer, gyroscope, and magnetometer) that can capture human motion data (e.g., moving speed and direction). Accelerometers measure the specific force across the x, y, and z axes in their local frame (see Figure 1). Gyroscope's angular velocity around the x, y, and z axes in their local frame. Magnetometers measure the Earth's magnetic field and provide the heading [22].

Traditional methods for step detection include peak detection, zero-crossing detection, and stance-phase detection [12]. These methods aim to detect a single step based on specific features [23, 29] in the sensor signal. However, previous works mainly focused on developing device-specific step detection algorithms in controlled experiment environments, which limits step count suitability across devices and user scenarios [19, 27]. Moreover, for better user experience, mobile devices usually require lower energy consumption and smaller-size algorithms for dealing with sequential sensor data.

TScIT 37, July 8, 2022, Enschede, The Netherlands

Deep learning approaches have been widely used in prediction and classification tasks across numerous domains, such as healthcare, visual recognition, audio processing, text analytics, and many more [28]. Recent works on CNN have shown good accuracy in step detection [8, 14, 19]. However, no related works are currently comparing different types of convolutions used in CNN model-based step detection through IMU data. This paper aims to compare the performance of convolution types in a lightweight CNN model that can be embedded into pedometer applications in mobile devices such as smartphones, smartwatches, and fitness trackers.

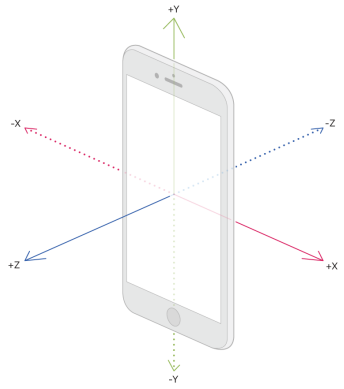

Fig. 1. Accelerometers measure changes in velocity along the x, y, and z axes, adopted from [3]

In the rest of this paper, we first discuss related work, introduce a common basic CNN architecture used in this research, describe the investigated convolution types, and then illustrate the experiments, including the dataset, experiment process, and evaluation measures. The paper ends with a discussion on the experiment results and future work.

2 RELATED WORK

Many research focused on providing specific high-performance CNN model implementations on step detection. For example, Lin [18] explored a machine learning-based approach for step detection and step counts in non-regular gait mode. The novelty is to analyze a window of time containing an arbitrary number of steps and integrate the detected count using a sliding window technique. Gamboa [8] proposed a novel unconstrained smartphone step detection model that utilizes the data from the accelerometer and gyroscope of the smartphone. This method uses the CNN model that contains five 1D layers with ReLU activation functions and one 1D layer with a sigmoid activation function as the output layer. Those research provide a good foundation for the experiment implementation for this research.

Other research focused on the comparisons or reviews of neural network architectures. For example, Ajit et al. [5] provided a

review of the CNN model layers explanation, how the CNN model works, and the different architectures of CNN models. Luu et al. [19], investigated and compared the accuracy of three typical neural network architectures for step detection in the clinical use case, i.e., shallow CNN, LSTM, and WaveNet. The results showed that it is possible to develop device-agnostic, accelerometer-only algorithms that provide highly accurate step count. Although this research's purpose is not to compare CNN architecture performance, previous studies of architecture specification and comparison provide valuable information for the CNN architecture selection for this research.

Previous studies of convolution types comparisons mainly fell on the research domain of sound and speech recognition. For example, Huang and Narayanan [11] investigated four types of convolutional operations by changing filter shape for speech emotion recognition to comprehensively understand the CNN model's performance. Krause et al. [16] compared different deep convolutional layers in CNN-based feature extraction for sound source localization. The convolution layers include commonly used 2D convolutions, 3D convolutions, and depthwise separable convolutions. Similar to [16], this paper investigates convolution layers on a common CNN architecture. In addition, we also investigate 1D convolutions and more convolution types under the 2D-based convolution type.

3 COMMON BASIC CNN ARCHITECTURE

The purpose of this study is not to propose optimized CNN architecture but instead to compare convolution types' influence on the performance of a CNN model in step detection. Thus, it is necessary to select an appropriate CNN architecture and use the same architecture to facilitate the comparison of different realizations of convolution.

Based on the study [19], we considered several high-performing neural network architectures which can be used for capturing time-series data features. Specifically, CNN [9], WaveNet [4] and LSTM [6]. We adopted the Shallow (or lightweight) CNN architecture as a common architecture for the experiments. Among those architectures mentioned above, the Shallow architecture has been proved to be more computationally effective, which could be useful in situations where algorithms deployed on the mobile device are limited by computational resources. [19]. Moreover, a shallow architecture with a minimal number of parameters (weights) is also able to avoid possible overfitting when the model is being trained [21]

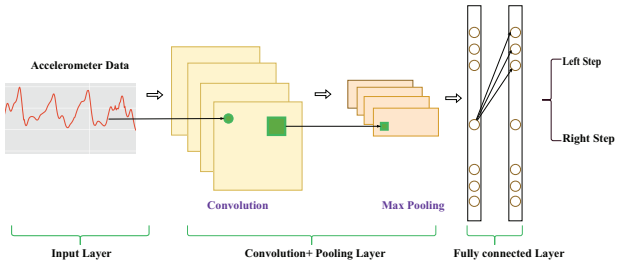

Fig. 2. Common shallow CNN model

The shallow CNN architecture (see Figure 2) in this research is composed of only four layers: 1 convolutional layer (of different convolution types), followed by 1 maxpooling layer and 2 fully connected layers. To be more specific,the model is composed of a single convolution layer, followed by a max-pooling layer and a 128-dimensional dense layer, then followed by a final 1-dimensianl dense (output) layer. Initially, the input data is reshaped according to the convolution layer. The convolution layer and the first dense layer had a Rectified Linear Unit activation function: $Relu(z) = max(0, z)$, where z is the input to a neuron. The final dense layer had a sigmoid activation function: $\sigma(z) = \frac{1}{1+e^{-z}}$,where Z is an element of input vector z.

4 CONVOLUTION TYPES TO BE INVESTIGATED

In a CNN model, the first and most important layer is a convolutional layer. It applies a convolution operation to the input, and pass the result to the next layer. The convolution operation uses multiple filters to extract input data features, and each filter uses a matrix called kernel to implement element-wise multiplication (or dot product) operations and output a represented matrix of input features as feature map. For example, in Figure 3, the input is a 2D matrix

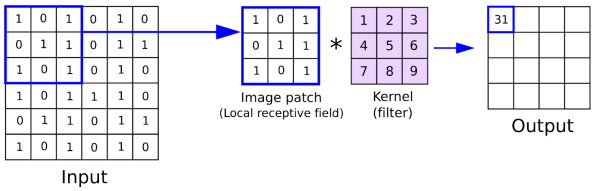

Fig. 3. Convolutional operation, adopted from [2]

[[1,0,1,0,1,0],[0,1,1,0,1,1],...,[1,0,1,0,1,0]], the filter kernel is also a 2D matrix [[1,2,3],[4,5,6],[7,8,9]], when the filter slides through the input data along width and height, it operates dot product between these two vectors, e.g., the output of 31 is calculated from input elements [[1,0,1],[0,1,1],[1,0,1]] in the top let corner, dot product by the kernel as:$1*1+0*2+1*3+1*4+1*5+1*6+1*7+0*8+1*9 = 31$.

Three hyperparameters have to be decided in convolution operations beforehand, including [1]: number of filters, stride, and padding. The number of filters affects the depth of the output since a single filter produces one feature map. If multiple filters are used, then the output matrix's depth will also grow. The stride is the distance the kernel escapes when moving along the input matrix. Padding is the extra data added to the input matrix when being processed by the kernel. For example, zero-padding adds 0 around the input matrix to keep the shape of the output the same as the input.

This section will introduce three commonly used convolution types, including 1D, 2D, and 3D. Meanwhile, I will further discuss four particular convolution types in the 2D CNN context, including depthwise separable, dilated, grouped, and transposed convolutions.

4.1 1D convolutions

Several research [17, 25] showed 1D-CNN works well with time-series sequence data. The kernel is a one-dimensional matrix. Each time step is associated with three featured accelerometer data (x,y,z). The kernel convolves along with the time series. Since the data is flattened to 1 channel, it will lack the information on the inter-channel dependencies.

4.2 2D Convolutions

A typical 2D convolution layer uses a two-dimensional kernel to process each input channel separately and sum the results over all the input channels for each filter. Although it might lack inter-channel relations, for example, the relation between accelerometer data in axial-x and axial-z, it is an efficient way to model features along with time-series data. In addition to regular 2D convolution, we investigated 2D-based depthwise separable, dilated, transposed, and grouped convolution.

Depthwise convolution is a spatial convolution performed independently over each input channel. It firstly applies a 2D kernel to each channel separately like a regular 2D convolution, then uses a 1D kernel to extract inter-channel dependencies and lower the channel dimension size to one [16].

Dilated convolution works the same as standard 2D convolution but with a broader kernel created by regularly inserting spaces/gaps between the kernel elements. For example, a dilution rate of 2 means the gap size is 1 in the kernel in the experiments.

Grouped convolutions were used in Alexnet so that a deep neural network can be trained on constrained devices with less powerful GPUs with smaller RAM [26]. A standard convolution layer convolves all of the input channels for each filter. In contrast, a grouped convolution with two groups only applies half of the filters to convolve half of the input channels. Thus for a number of groups x, the parameter cost is reduced by a factor of x.

Transposed convolution applies the same calculation as standard 2D convolution, except it uses a kernel to up-sample the padded input data.

4.3 3D Convolutions

Similar to 2D convolution, a 3D convolution uses a three-dimensional kernel to convolve input data. The filter slides through 3 dimensions of input data (width, height, channels) and generates a 3D matrix. It allows the network to learn inter-and intra-channel features simultaneously.

5 EXPERIMENTS

5.1 Dataset

5.1.1 Public Dataset. Experiments are performed using the public dataset for pedometer application [20]. Thirty participants performed three walking activities wearing Shimmer3 devices at the hip, ankle, and wrist in different gait modes: walking around a track with a regular and consistent gait, walking through a building, and moving around a room with varying amounts of pauses and gait changes. Meanwhile, the activities are video recorded with an iPhone to label whether each time step corresponded to the left or right step as ground truth data.

For this study, we only used regular walking activity and tri-axial accelerometer signal (see Figure 4) from the wrist that was sampled at 15 Hz. IMU-based step detection has shown that only the use of accelerometer data (and not gyroscope data) could significantly reduce battery consumption, higher integrity of recorded data, and even higher accuracy in model testing [8, 19].

	timestamp	x	y	z	label
0	1.466788e+09	0.658020	0.418594	-0.226257	NaN
1	1.466788e+09	0.023987	0.424332	-0.424988	NaN
2	1.466788e+09	-0.309265	1.292130	-0.924255	NaN
3	1.466788e+09	0.096130	1.440933	-0.527405	NaN
4	1.466788e+09	0.216248	1.220963	-0.021301	r

Fig. 4. Accelerometer dataset

In the end, each time step in the raw data which will be used in the experiment contains four variables: three variables from accelerometer (x,y,z) followed by a ground truth label.

5.1.2 Data Pre-processing. We have to pre-process the input data before we feed them into a CNN model. Firstly, the data format has to be compatible with a specific initial convolutional layer of the model architecture; secondly, the data is recommended to be normalized and standardized for better model performance; thirdly, data features are extracted when IMU data is divided into data blocks (or windows).

Before the model training, we partitioned each raw data series into windows of 2s seconds of data (or 30 time steps), and then extracted and labeled each window feature as left/right step and encoded left as 1 and right as 0 accordingly. Multiple ways have been used for step detection label.For example, a typical way for step classification is to use the most frequent occurrence of left/right step type as label [24]. Another way is to identify whether a data frame is a step or not without differentiating the left/right step type. Different from the previous study,Luu et al. [19] used the last time step type within a window as the label and changed the classification problem into a left/right step prediction based on past signals. For step counts, most researchers use ground truth data within a window to identify the number of steps as the label and use the model to predict an unknown window [18, 19]. For this research, to train and test the model accuracy, the last timestamp step type within a window is used as the label for step classification.

5.2 Model Training, Validation and Testing

The cross-validation of public data is done by splitting it into training and validation, testing in a ratio of 8:1:1. The main idea for the experiments is to compare convolution types in a typical CNN architecture. Thus once replace the convolution type in the architecture, each model will be trained, validated, and tested in the same way.

For each model with different convolution types, the same experiment process is used (see Figure 5). When the investigated convolution type is changed in this common architecture, we reshaped the window accordingly (see Table 1). Except for the shape of input data, other parameters are kept the same under the common

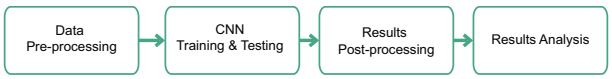

Fig. 5. Experiment process

shallow CNN architecture. The training for each model includes ten iterations in which 1/9 of the training dataset is shuffled and used as the validation data and then tested with the same test dataset. The output of the results from the model is a series of data between 0 and 1, and we used 0.5 as the threshold value to identify the results as 0 and 1, and compared with the actual steps (left step is label as 1, right step is labeled as 0).

Table 1. Input data shape

CNN	Input shape
1D	(30,3)
2D	(10,3,3)
3D	(5,3,2,3)

We implemented the shallow model using Python Keras [15] libraries with TensorFlow in Jupyter Notebook. All the models are compiled and fit with the same setting, and the batch size was 250. The optimizer was Adam with default values (learning rate=0.001, beta 1=0.9, beta 2=0.999, epsilon=1e-07, amsgrad=False), the loss function was binary-entropy, the number of epochs was 100. Early stopping with a patience of 10, and the validation loss was used as the monitor.

5.3 Evaluation Measures

This research used two main evaluation measures, including accuracy and model complexity. A typical step detection problem includes a step counts task, i.e., counting the total number of steps within an arbitrary time window; and a step classification task, i.e., given a window of time-serial IMU datasets and identifying whether a step happened or not, or classify left or right step. This research mainly focuses on the accuracy of the step classification task by identifying the left and right steps. Since step classification can be a foundation for other applications such as step counts [19] and PDR-based navigation [23]. The accuracy for step classification is calculated as follows:

$$Accuracy = \frac{n\_correct}{N\_total}$$

Where n_correct is the number of correctly classified steps and N_total is the total number of steps.

The computational complexity is defined as the number of trainable network parameters which can be generated by the Keras model summary function. Pragmatically, a model with higher complexity indicates higher computation cost, resulting in longer training and testing time.

6 RESULTS

An average accuracy of 60% [24] and 76% [19] of step classification were achieved respectively in previous works. The shallow CNN model in this research has achieved a result close to [24]. Since step classification is harder to achieve in contrast to step counts accuracy, which most research can achieve about 90% or higher [8, 18, 19]. We achieved relatively low accuracy in this research because we tried to classify the left and right steps when the signal features of the left and right steps are highly similar.

Table 2. Comparison of different dimensions of convolution types

CNN	Complexity	Accuracy
1D	69,737	66.9%
2D	24,305	63.7%
3D	12,425	65.0%

In the results of dimensional convolutions comparison (see Table 2), the 1D convolution shows an accuracy of 66.9% with the highest complexity among different dimensional convolutions, while 3D convolution results in the lowest complexity (about 1/5 of 1D convolution) and accuracy. The accuracy difference among different dimensional convolutions is about 2%. Contrary to typical CNN classification research [16, 30], the 3D convolution usually has the highest complexity among dimensional convolutions. This research achieved the lowest complexity concerning 3D convolutions. Because in this research, we kept the same input time steps in each window and channel depth as three and then reshaped each window, leading to a smaller dimensional size of input data for 3D convolution (see Table 1).

In the context of the 2D shallow model (see Table 3), the accuracy difference (around 1%) among convolution types is slight. However, depthwise separable convolution shows the smallest complexity, almost 1/10 of other convolution types. This result is similar to [16] which stated that depthwise separable convolution achieved the lowest complexity in the application of CNN for sound source localization. This could be explained by the final step of depthwise separable convolution. It stacks each channel and reduces the channel dimension size to one. As a result, it reduces the parameter numbers and extracts inter- and intra-channel features by filtering each channel separately first, then combining them.

Table 3. Comparison of 2D-based convolution types

CNN	Complexity	Accuracy
2D-Depthwise	2,207	59.8%
2D-Dilated	24,305	61.7%
2D-Transposed	24,305	62.3%
2D-Grouped	23,657	58.7%

7 CONCLUSIONS

This research is a comparative study of different convolutional layers in a shallow CNN model for step detection. We evaluated the accuracy and model complexity in model performance using only regular walking accelerometer data from a public dataset. 1D convolutions showed a relatively higher accuracy at a high computational cost. Although 3D convolution gets a smaller complexity result compared with 1D and 2D in this experiment, the memory requirement

might be higher than other convolution types. In the 2D convolutions context, the depthwise separable convolutions achieved good accuracy with the lowest complexity among all convolution types. It brings two benefits: fewer parameters are less prone to overfitting and less computation time, which means a faster and cheaper model.

In order to provide a more thorough comparison of convolution types, future work should consider more evaluation measures to provide better supporting work for its performance in mobile devices, such as memory usage, inference time, and energy consumption. In addition to testing the model's performance in a regular walking scenario, also test irregular walking with different devices (e.g., smartphone and smartwatch) in different body locations (e.g., front/back pocket, in the bag, and hand). Lastly, besides the public dataset, a self-collected independent dataset should be added to test the robustness and generalizability of models.

REFERENCES

[1] 2020. https://www.ibm.com/cloud/learn/convolutional-neural-networks

[2] 2021. https://analyticsindiamag.com/what-is-a-convolutional-layer/

[3] 2022. https://developer.apple.com/documentation/coremotion/getting_raw_accelerometer_events

[4] Aaron, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. https://arxiv.org/abs/1609.03499

[5] Arohan Ajit, Koustav Acharya, and Abhishek Samanta. 2020. A Review of Convolutional Neural Networks. In *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*. https://doi.org/10.1109/ic-etite47903.2020.049

[6] Shaojie Bai, Kolter J Zico, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. https://arxiv.org/abs/1803.01271

[7] Agata Brajdic and Robert Harle. 2013. Walk detection and step counting on unconstrained smartphones. *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13* (2013). https://doi.org/10.1145/2493432.2493449

[8] Hugo Filipe Silveira Gamboa. 2020. *Real-Time Step Detection Using Unconstrained Smartphone*. Ph. D. Dissertation.

[9] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional Sequence to Sequence Learning. *arXiv:1705.03122 [cs]* (Jul 2017). https://arxiv.org/abs/1705.03122v3

[10] Alexander Hodkinson, Evangelos Kontopantelis, Charles Adeniji, Harm van Marwijk, Brian McMillan, Peter Bower, and Maria Panagioti. 2019. Accelerometer- and Pedometer-Based Physical Activity Interventions Among Adults With Cardiometabolic Conditions. *JAMA Network Open* 2, 10 (Oct 2019), e1912895. https://doi.org/10.1001/jamanetworkopen.2019.12895

[11] Che-Wei Huang and Shrikanth S. Narayanan. 2018. Characterizing Types of Convolution in Deep Convolutional Recurrent Neural Networks for Robust Speech Emotion Recognition. *arXiv:1706.02901 [cs]* (Jan 2018). https://arxiv.org/abs/1706.02901

[12] A.R. Jimenez, F. Seco, C. Prieto, and J. Guevara. 2009. A comparison of Pedestrian Dead-Reckoning algorithms using a low-cost MEMS IMU. , 37–42 pages. https://doi.org/10.1109/WISP.2009.5286542

[13] Noriaki Kakiuchi and Shunsuke Kamijo. 2013. Pedestrian dead reckoning for mobile phones through walking and running mode recognition. *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)* (Oct 2013). https://doi.org/10.1109/itsc.2013.6728243

[14] Xiaomin Kang, Baoqi Huang, and Guodong Qi. 2018. A Novel Walking Detection and Step Counting Algorithm Using Unconstrained Smartphones. *Sensors* 18, 1 (Jan 2018), 297. https://doi.org/10.3390/s18010297

[15] keras team. 2019. keras-team/keras. https://github.com/keras-team/keras

[16] Daniel Krause, Archontis Politis, and Konrad Kowalczyk. 2021. Comparison of Convolution Types in CNN-based Feature Extraction for Sound Source Localization. *2020 28th European Signal Processing Conference (EUSIPCO)* (Jan 2021), 820–824. https://doi.org/10.23919/eusipco47968.2020.9287344

[17] Moshe Kravchik and Asaf Shabtai. 2018. Detecting Cyber Attacks in Industrial Control Systems Using Convolutional Neural Networks. *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy* (Jan 2018). https://doi.org/10.1145/3264888.3264896

[18] Basil Lin. 2020. *Machine Learning and Pedometers: An Integration-Based Convolutional Neural Network for Step Counting and Detection*. Ph. D. Dissertation.

[19] Long Luu, Arvind Pillai, Halsey Lea, Ruben Buendia, Faisal M. Khan, and Glynn Dennis. 2021. Accurate Step Count With Generalizable Deep Learning on Accelerometer Data. *2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)* (Oct 2021). https://doi.org/10.1109/dasc-picom-cbdcom-cyberscitech52372.2021.00042

[20] Ryan Mattfeld, Elliot Jesch, and Adam Hoover. 2017. A new dataset for evaluating pedometer performance. *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* (Nov 2017). https://doi.org/10.1109/bibm.2017.8217769

[21] Himadri Mukherjee, Subhankar Ghosh, Ankita Dhar, Sk. Md. Obaidullah, KC Santosh, and Kaushik Roy. 2020. Shallow Convolutional Neural Network for COVID-19 Outbreak Screening using Chest X-rays. (Apr 2020). https://doi.org/10.36227/techrxiv.12156522

[22] Barak Or. 2021. What is IMU? https://towardsdatascience.com/what-is-imu-9565e55b44c

[23] So Young Park, Se Jong Heo, and Chan Gook Park. 2017. Accelerometer-based smartphone step detection using machine learning technique. *2017 International Electrical Engineering Congress (iEECON)* (Mar 2017). https://doi.org/10.1109/ieecon.2017.8075875

[24] Arvind Pillai, Halsey Lea, Faisal Khan, and Glynn Dennis. 2020. Personalized Step Counting Using Wearable Sensors: A Domain Adapted LSTM Network Approach.

[25] Masoumeh Rahimi, Alireza Alghassi, Mominul Ahsan, and Julfikar Haider. 2020. Deep Learning Model for Industrial Leakage Detection Using Acoustic Emission Signal. *Informatics* 7, 4 (Nov 2020), 49. https://doi.org/10.3390/informatics7040049

[26] Sabyasachi Sahoo. 2018. Grouped Convolutions — convolutions in parallel. https://towardsdatascience.com/grouped-convolutions-convolutions-in-parallel-3b8cc847e851

[27] Dario Salvi, Carmelo Velardo, Jamieson Brynes, and Lionel Tarassenko. 2018. An Optimised Algorithm for Accurate Steps Counting From Smart-Phone Accelerometry. , 4423–4427 pages. https://doi.org/10.1109/EMBC.2018.8513319

[28] Iqbal H. Sarker. 2021. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science* 2, 6 (Aug 2021). https://doi.org/10.1007/s42979-021-00815-1

[29] Xiaokun Yang and Baoqi Huang. 2015. An accurate step detection algorithm using unconstrained smartphones. *The 27th Chinese Control and Decision Conference (2015 CCDC)* (May 2015). https://doi.org/10.1109/ccdc.2015.7161816

[30] Duona Zhang, Wenrui Ding, Baochang Zhang, Chunyu Xie, Hongguang Li, Chunhui Liu, and Jungong Han. 2018. Automatic Modulation Classification Based on Deep Learning for Unmanned Aerial Vehicles. *Sensors* 18, 3 (Mar 2018), 924. https://doi.org/10.3390/s18030924