Leite, Clayton Frederick Souza; Xiao, Yu

Optimal sensor channel selection for resource-efficient deep activity recognition

Published in:
Proceedings of the 20th International Conference on Information Processing in Sensor Networks, IPSN 2021

DOI:
10.1145/3412382.3458278

Published: 18/05/2021

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Leite, C. F. S., & Xiao, Y. (2021). Optimal sensor channel selection for resource-efficient deep activity recognition. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks, IPSN 2021* (pp. 371-383). Article 3458278 ACM. https://doi.org/10.1145/3412382.3458278

Optimal sensor channel selection for resource-efficient deep activity recognition

Clayton Frederick Souza Leite
clayton.souzaleite@aalto.fi
Department of Communications and Networking
Aalto University

Yu Xiao
yu.xiao@aalto.fi
Department of Communications and Networking
Aalto University

ABSTRACT

Deep learning has permitted unprecedented performance in sensor-based human activity recognition (HAR). However, deep learning models often present high computational overheads, which poses challenges to their implementation on resource-constraint devices such as microcontrollers. Usually, the computational overhead increases with the input size. One way to reduce the input size is by constraining the number of sensor channels. We refer to sensor channel as a specific data modality (e.g. accelerometer) placed on a specific body location (e.g. chest). Identifying and removing irrelevant and redundant sensor channels is feasible via exhaustive search only in cases where few candidates exist. In this paper, we propose a smarter and more efficient way to optimize the sensor channel selection during the training of deep neural networks for HAR. Firstly, we propose a light-weight deep neural network architecture that learns to minimize the use of redundant and irrelevant information in the classification task, while achieving high performance. Secondly, we propose a sensor channel selection algorithm that utilizes the knowledge learned by the neural network to rank the sensor channels by their contribution to the classification task. The neural network is then trimmed by removing the sensor channels with the least contribution from the input and pruning the corresponding weights involved in processing them. The pipeline that consists of the above two steps iterates until the optimal set of sensor channels has been found to balance the trade-off between resource consumption and classification performance. Compared with other selection methods in the literature, experiments on 5 public datasets showed that our proposal achieved significantly higher F1-scores at the same time as utilizing from 76% to 93% less memory, with up to 75% faster inference time and as far as 76% lower energy consumption.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Sensor channel selection; human activity recognition; deep learning

IPSN' 21, May 18–21, 2021, Nashville, TN, USA

ACM ISBN 978-1-4503-8098-0/21/05…$15.00
https://doi.org/10.1145/3412382.3458278

ACM Reference Format:
Clayton Frederick Souza Leite and Yu Xiao. 2021. Optimal sensor channel selection for resource-efficient deep activity recognition. In *Information Processing in Sensor Networks (IPSN' 21), May 18–21, 2021, Nashville, TN, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3412382.3458278

1 INTRODUCTION

Human activity recognition (HAR) using wearable sensors has numerous applications in healthcare [23], smart home systems [7], sign language translation [4], virtual reality interaction [14], and etc. Taken the readings of multiple **sensor channels** as input, the algorithms of HAR automatically identify the type of actions an individual is performing. We refer to **sensor channel** (or simply channel) as a stream of data of a specific modality (e.g. heart rate meter, accelerometer, gyroscope, magnetometer, etc.) placed on a specific part of the user's body. For instance, an accelerometer placed on the left wrist, an accelerometer on the chest, and a magnetometer on the right ankle represent three different sensor channels.

Among the methods for HAR, deep learning (DL) has been delivering impressive performance without the need for domain knowledge for feature engineering. However, the utilization of DL in real-time requires large amounts of available memory and computational power, considerably beyond the specifications of common low-cost off-the-shelf microcontrollers [25]. The computational overhead typically increases with the number of sensor channels involved in the input. A straightforward way to reduce resource consumption is to constrain the number of sensor channels without significant performance degradation. **Sensor channel selection** is defined as the identification and removal of channels that provide a negative, null, or negligible contribution to the performance of a HAR classifier. It shares the same goal with neural network compression techniques. The difference is that, in compression techniques, the aim is to minimize the neural network complexity, whereas sensor channel selection targets the minimization of the input complexity.

Exhaustive search is a commonly used approach for sensor channel selection that trains neural networks with all possible combinations of channels as input and selects the one that fulfills the performance requirements with fewer channels. This approach is not scalable, since the number of possible sensor channel combinations increases exponentially as the number of sensor channels increases, thus leading to extremely long training duration - especially in DL scenarios. In this paper, we propose a pipeline that is able to spot and select relevant and non-redundant sensor channels more efficiently during the training process of the deep neural network-based activity classifier.

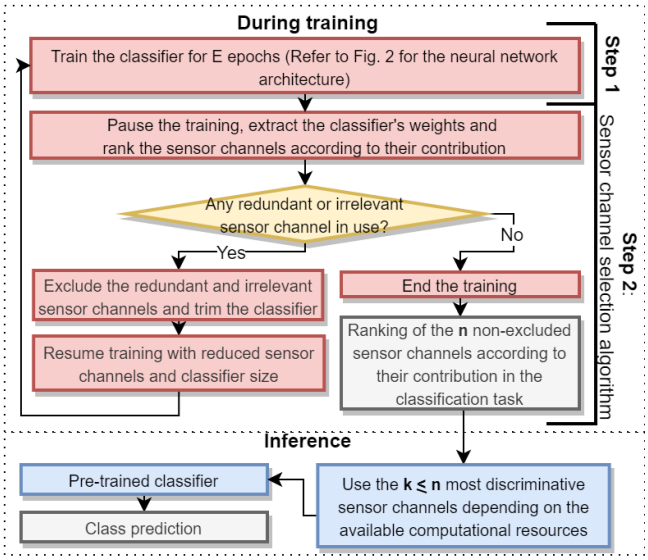

Figure 1: The pipeline of our method. Firstly, the classifier is trained for a specific number of epochs E defined by the designer. Next, the training is paused while the sensor selection algorithm searches for and removes redundant or irrelevant sensor channels. The classifier training resumes with fewer weights and fewer channels. At the end of the training, our solution provides a list with all the n remained channels ranked according to their contribution to the classification. At run-time, the $k \leq n$ most discriminative channels are utilized, where k depends on the computational budget.

Sensor channel selection has also been commonly approached by training a separate classifier for each channel, fusing the predictions of each classifier into a final one, and then removing classifiers (and hence channels) while minimizing the impact on the final prediction. Since each classifier is trained independently of the others, this technique doesn't allow for the learning of inter-channel dependencies. That is, intricate relationships across sensor channels are not learned, which leads to worsened performance. Our method allows for inter-channel dependency learning by not splitting the channels during training and at inference time.

In shallow learning scenarios, feature selection may lead to sensor channel selection. For instance, if a feature selection algorithm opts to exclude all the features corresponding to a certain sensor channel, the sensor channel in question can also be excluded. In DL, feature selection cannot be employed to filter out sensor channels. This is because the features learned after multiple stacked layers in a deep neural network typically contain simultaneously bits of information from all sensor channels in an unpredictable manner. Hence, even if feature selection is to be applied in DL, the selected features will still include information from all channels. Moreover, DL models are a black box with respect to how the information from each of the available sensor channels is utilized to generate a prediction. This characteristic poses a challenge to identifying redundant and irrelevant sensor channels and, consequently, makes the problem of sensor channel selection substantially hard.

As illustrated in Fig. 1, we solve the above-mentioned challenges in a two-stage iterated process. The key features of the pipeline, as well as the key contributions of this paper, are summarized as follows:

- A novel and light-weight deep neural network architecture for sensor-based HAR. It is trained with an objective function that instructs it to learn to minimize the use of redundant and irrelevant information across sensor channels while maintaining high performance compared to other approaches and without resorting to an exhaustive search.
- A more efficient sensor channel selection algorithm that looks into the neural network weights, interprets them to decipher which channels are more discriminative and identifies the redundant and irrelevant ones to the classification task. Moreover, this algorithm trims the neural network leaving only the part responsible for processing information from the relevant and non-redundant sensor channels. This leads to a significant reduction of resource utilization during the inference.

For evaluation, we conducted experiments with 5 public datasets: PAMAP2 [22], Opportunity [6], Skoda [32], MHEALTH [3], and Daphnet [2]. Compared with other selection methods in the literature [5, 15, 17, 30], our proposal achieved significantly higher F1-scores at the same time as utilizing from 75% to 93% less memory, with up to 75% faster inference time and as far as 76% lower energy consumption. Despite our pipeline having been proposed for HAR, we believe that it can be successfully utilized to other multi-input problems in DL since the methods therein are general.

The rest of this work is organized as follows. In Section 2, we describe the related work. In Section 3, our pipeline is explained. The experimental setup and results are detailed in Section 4 and Section 5, respectively. Section 6 discusses the advantages and limitations of our proposal, including future work, followed by the conclusions in Section 7.

2 RELATED WORK

We review in this section previous works on the sensor channel selection using exhaustive search and optimization-based methods, respectively. Table 1 provides an overview of the features of our pipeline compared to the literature.

2.1 Exhaustive search

An exhaustive search algorithm consists in testing all (or a subset of all) possible combinations of sensor channels, as opposed to an optimization-based algorithm, where the selection of sensor channels is obtained by optimizing an objective function. An exhaustive search is only plausible in cases where the number of channels is small enough to permit only very few different combinations of them. For instance, a number of channels above 5 generate more than 32 possible combinations, which renders the exhaustive search prohibitively expensive, especially if it consists in training DL models. To reduce the computational cost, a simple way is to shrink the search space by performing the search on only a subset of all possible combinations. However, if it is done randomly, it can lead to sub-optimal solutions. To avoid this, heuristics can be used

Feature	Ours	Exhaustive search [1, 8, 9, 30]	Sensor channel selection [5, 15, 17, 29]	Standard DL methods [10, 11, 19, 20, 28]
Classification performance	**High**	Medium-High	Medium	High
Resource utilization	**Lower**	Low	Low - Medium	High
Redundant and irrelevant channels are removed	**Yes**	Yes	Yes	No
Rank channels by their importance	**Yes**	No	No	No
Duration of the training	Medium-Long	Extremely long	Long	**Medium**

Table 1: Comparison of our solution with the related work.

in the shortening of the search space, but it may require domain knowledge.

Aziz *et al.* [1] determined the minimum amount of body-worn IMU sensors and their locations to accurately classify the daily living activities including walking, sitting down, standing up, and standing still. Their method consisted in exhaustively testing 8 different settings - each setting with a certain number of sensor channels. The setting that provided the best performance was formed by 3 IMU sensors, one on each of the following locations: chest, right ankle, and left ankle. Similar work with an exhaustive search for daily living activities was performed by Ertugrul and Kaya [8]. In [9], an exhaustive search composed of $2^{14} - 1$ combinations of sensor channels was performed with shallow learning for the recognition of home activities.

The approach in [30] consisted of having a classifier for every individual channel. To fuse the predictions of all classifiers into a unique class, a Naïve Bayes classifier was used. The authors focused on the case where at run-time one or multiple sensor channels may become unavailable (e.g. due to malfunction). During the training, the authors then tested 3800 different combinations of sensor channels, saving in memory the statistics of the performance of each combination for later reference. At run-time, a sensor channel selection algorithm was used to pick a minimal and best-performing combination depending on the availability of the sensor channels. Follow-up work was presented in [31], where the authors replaced the exhaustive search during model training with a more efficient heuristics at run-time that reduced the search space from $2^N - 1$ possibilities to $2^K - 1$, where $K \leq N$ and N is the number of sensor channels.

Compared to shallow classifiers, which are utilized in the aforementioned works, DL models have a more complex structure with a number of trainable variables that can reach millions. Moreover, the designer of a DL model often has to go through the process of selecting optimal values for the hyper-parameters by trial and error, which signifies training the model multiple times. Therefore, in the case of DL, exhaustive searches are expensive to an even greater extent. For this, our proposal does not employ an exhaustive search.

2.2 Optimization-based methods

Yang *et al.* [29] proposed a sensor channel selection algorithm for shallow learning. The authors addressed the problem by optimizing the performance and number of selected channels via Markov Decision Process. Their algorithm is composed of two shallow neural networks - a classification network and a policy network. At a certain instant of time, all channels are sampled and 13 different handcrafted features are extracted from their data. These features are fed into the policy network, which provides as output a vector of binary values that represents the state of each sensor channel (selected or not selected). The extracted features suffer a sparsification: features of sensor channels that have not been selected are set to zeros, whereas those of selected channels are left unchanged. The classification network receives the sparse features as input and classifies them according to the activity.

Two important points should be clarified about Yang's [29] method. First, a reduction in computational expense only happens if the cost corresponding to the policy network plus the cost involved in the classification of the sparse features does not surpass the cost of classifying the dense features. The authors did not quantify these costs. However, performing computations on sparse features may not lead to an appreciable reduction in computational expense, unless the device responsible for the computations is powered with accelerations for sparse tensor algebra. This is hardly available in microcontrollers or microprocessors, thus limiting the lightness of their method with respect to the computational device. Second, the memory footprint of the classification model is not reduced by the fact that it receives sparse input data. On the contrary, there is an increase in this factor by the use of a secondary network - the policy network.

We devise an optimization-based method that presents modularity with respect to the sensor channels - as it is also done in [29]. However, differently, our method does not require zeroing-out inputs corresponding to excluded sensor channels. Therefore, in this sense, gains in resource efficiency are more obtainable since sparse tensor algebra is not required. Also differently from [29], memory footprint in our neural network can be reduced when sensor channels are excluded.

In Cao *et al.* [5], similar to [30] and [31], there are as many classifiers as sensor channels - with each sensor channel connected to a corresponding classifier. The authors then transformed the problem of sensor channel selection into classifier selection and proposed two distinct methods to select a subset of classifiers. For instance, one of the proposed methods (mRMR) quantifies redundancy among classifiers in order to select a subset of classifiers with minimal redundancy.

Keally *et al.* [15] also resorted to ensemble learning (training multiple classifiers, each of which assigned to a channel). However, to identify redundancies across channels, the authors proposed to use the Pearson correlation coefficient. In [17], device selection - where each device is composed of a group of channels and has its individual classifier - was addressed. In their method, at every k seconds, a single device among all the available ones is selected based on metrics that aim at quantifying its quality (by quality it is meant capability of generating accurate predictions). The selected device is then used to generate predictions until a new selection is performed.

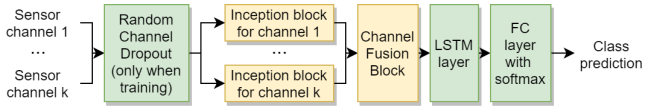

Figure 2: Our neural network architecture designed for resource-efficient HAR. Layers are shown in green, while blocks appear in yellow. Our neural network is characterized by the presence of a dropout layer at the beginning, which randomly drops sensor channels. A separated inception block performs the learning of short-temporal and spatial dependencies within each channel. In the next stage, the channels are fused according to their relevancy to the classification. This is where spatial dependencies between the channels are learned. The long short-term memory (LSTM) layer learns long-temporal dependencies in the data. Finally, the fully-connected (FC) layer activated by a softmax function provides per-class predictions.

A drawback of having multiple classifiers for each channel - or for each device as in the case of [17] - is that the classifiers do not learn relationships across the sensor channels, which may lead to reduced performance. By avoiding assigning a separate classifier for each sensor channel and by combining features originated from different channels before the classification layer, our neural network is able to learn inter-channel dependencies.

2.3 Controlling the number of selected sensor channels at run-time

In scenarios where the levels of resources - such as processing power, and memory availability - are dynamically changing, controlling the number of selected channels at run-time is an important feature that allows the HAR system to adapt to these varying conditions. In the optimization-based methods discussed [5, 29], this feature is not present. In [29], alternating between a more cost-saving approach (i.e., with fewer selected channels) and a more accuracy-oriented approach (i.e., with a greater number of selected channels) is only possible through complete retraining of both networks. In [5], this feature can be implemented at run-time without the need for retraining. However, for this, modifications to the method that selects the classifiers are necessary. Complete retraining of the classifiers is also necessary for the exhaustive search methods discussed, with the exception of [30], whose system was specifically designed for this purpose.

In our proposal, when the training is complete, our sensor channel selection algorithm provides an ordered list of channels according to their contribution. At run-time, a subset of these channels can be selected. Such a subset can have variable size depending on the level of resource availability at the moment.

3 METHODS

We propose a neural network architecture, as illustrated in Fig. 2, for predicting the class of human activities based on sensory input. Our design described in this section follows the principle of minimizing the use of redundant and irrelevant information across sensor channels, without causing significant performance degradation. A sensor channel is denoted as $s_i \in \mathbb{R}^a \times \mathbb{R}^b$ with i as its identifier. a and b are, respectively, the time-length and spatial length of a sliding window. For instance, using the 3D accelerometer (acceleration in x, y, and z axes) as an example, $b = 3$. Each component of the network is described as follows.

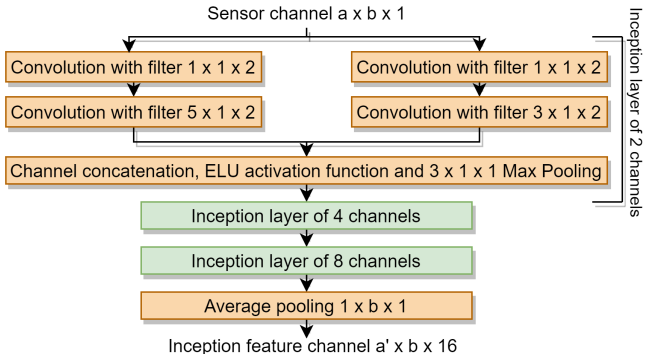

Figure 3: The inception block. Dimensions a and b represent, respectively, the sliding window's time-length and spatial-length. For instance, a sliding window of 100 time-samples from a 3D accelerometer has dimensions 100 x 3. The first dimension a is pooled with kernel size 3, resulting in $a' = \text{ceil}(a/27)$ at the output.

3.1 Dropout layer

The neural network starts with a dropout layer, originally proposed in [26]. When training, the dropout layer removes a sensor channel i with a pre-defined probability p, which is constant and common to all sensor channels. Consequently, the corresponding data streams are removed from the input of the following layers. In other words, they will not be processed in the following layers. We denote as o_i the state of the sensor channel i, in the sense that $o_i = 0$ if s_i was dropped out - and otherwise $o_i = 1$.

The dropout layer was originally proposed to avoid overfitting in neural networks by breaking up co-adaptations specific to the training data, thus allowing the network to generalize better on unseen data [26]. Similarly, we base on the premise that standard back-propagation builds co-adaptations between sensor channels. This makes the trained network highly dependent on all sensor channels. Even if some channels do not contribute, the performance of the network would degrade if any of them is to be replaced with random noise or zeros. By using dropout on sensor channels, the network is enforced to learn to work with random combinations of channels.

Higher values of p drop channels more often, thus slowing down the learning of the neural network - i.e. decreasing the convergence speed of the training. Smaller values of p defeat the purpose of channel selection since dropping out channels during training is important to break the co-adaptations between them. Therefore, p is a hyper-parameter that is tuned with the use of the results on the validation set.

3.2 Inception blocks

The sensor channels that were not dropped out in the previous layer are processed in inception blocks (Fig. 3). There is an independent inception block for each sensor channel i and we denote it as a

function $I_i(\cdot)$. The output of an inception block is written as $I_i(s_i) = f_i$. The inception block is originally proposed in [27] and has been shown to be effective for HAR in [28]. Our variant of the inception block is lighter and contains only 780 trainable variables. Fig. 3 illustrates the components of our inception block: three inception layers and an average pooling layer at the spatial dimension b. The goal of the inception block is to learn short-temporal and short-spatial dependencies present in the sensor channel. Long-spatial (i.e. across sensor channels) and long-temporal dependencies are learned by the subsequent channel fusion layer and long short-term memory (LSTM) layer.

3.3 Fusing channels

The feature maps extracted by the inception blocks - f_i - are input for the channel fusion layer. The fusion of the channels is made by performing a simple weighted sum of the normalized inception feature maps (Eq. 1), which collaborates with the goal of a light-weight model.

Some of the feature maps may have a different amplitude compared to the others. This imbalance harms our goal of having weight h_i that dictates the importance of its feature map f_i for accurate classification of the activities. For this reason, we normalize the feature maps. The calculation of the mean and standard deviation is shown in Eq. 2, which follows [16].

$$\sum_i h_i \cdot \frac{f_i - \mu_i}{\sigma_i}, \tag{1}$$

$$\mu_i = \frac{1}{U}\sum_j^U u_j^i \quad \text{and} \quad \sigma_i^2 = \frac{1}{U}\sum_j^U (u_j^i - \mu_i)^2, \tag{2}$$

where u_j^i is a feature present in f_i and U is the number of features in f_i. Notice that each feature map is normalized independently. Therefore, changes in one feature map do not produce correlated changes in the others.

The calculation of h_i is shown in Eq. 3.

$$h_i = \frac{o_i \cdot exp(v_i)}{\sum_j o_j \cdot exp(v_j)} \quad \text{thus} \quad \sum_i h_i = 1 \tag{3}$$

The variables v_i - sensor channel values - are calculated by Eq. 4.

$$v_i = \sum_{j, j\neq i} o_j \cdot w_{i|j} \tag{4}$$

The variable $w_{i|j}$ - denoted as cross-channel weight - is defined as the importance of the sensor channel i, given that the sensor channel j (with $j \neq i$) is also used for the classification. On the other hand, negative values for $w_{i|j}$ indicate the redundant or irrelevant nature of channel i in the presence of j. Each variable $w_{i|j}$ is a trainable variable and we impose the relationship $w_{i|j} = -w_{j|i}$, which enforces the network to learn the imbalance of the importance of sensor channels. The value of a cross-channel weight is only considered in v_i if its channel was not previously removed by the dropout layer. This is expressed by the multiplication with o_j. Moreover, h_i is not influenced by sensor channels that have been dropped out.

The sum of the channel weights h_i being equal to 1 and the utilization of the normalization layer gives the neural network a modular characteristic with respect to the sensor channels. The number of weights in the channel fusion block is as low as $N * (N-1)/2$, where N is the number of channels. This collaborates with the light-weight purpose of our neural network. The output of the channel fusion block is fed into the LSTM [13] layer, where long-term temporal dependencies within the fused channels are learned from the data.

3.4 Classification layer

The output of the LSTM layer is fed into a fully-connected (FC) layer which generates a vector with a size equal to the number of classes. A softmax layer proceeds to generate class probabilities. The loss function $\mathcal{L}$ to be minimized is composed of three distinct terms: 1) the standard cross-entropy for multiclass classification, 2) an entropy on the vector $\mathbf{h}$, and 3) a knowledge distillation cross-entropy objective as in [12]. The importance of each term is determined by the positive real-valued constants λ_C, λ_E and λ_D defined prior to the training.

$$\begin{aligned}\mathcal{L} = \mathbb{E}[\lambda_C \frac{\sum_i o_i}{N} \sum_k y_k \log(\hat{y}_k) - \lambda_E \sum_i h_i \log(h_i) \\ + \lambda_D \frac{\sum_i o_i}{N} \sum_k y'_k \log(\hat{y}_k)]\end{aligned} \tag{5}$$

In the first term - the cross-entropy with real labels - N represents the number of sensor channels. y_k and $\hat{y}_k$ represent, respectively, the label and the prediction probability for class c. The multiplying term $\frac{\sum_i o_i}{N}$ is a way of adjusting the gradients with the number of sensor channels that were not dropped out. Therefore, bigger changes in the values of the network's weights come from cases where fewer sensor channels have been dropped out.

Note that the weights $w_{i|j}$ are trained like any other parameter in the neural network. However, minimizing the entropy of the vector $\mathbf{h}$ - the second term - forces an imbalance in these weights, which stimulates the network to learn to utilize as few sensor channels as possible for accurate classification. The negative sign before the entropy term and a positive real-valued constant λ_E dictate the minimization of the entropy objective.

Regarding the third term, i.e. the knowledge distillation term, y'_k is the prediction - named soft labels - given by a teacher model and is defined in Eq. 6. The teacher model is a fairly more complex neural network - as InnoHAR [28] - utilizing all sensor channels without dropout. Preliminary experiments performed by us have shown that distilling knowledge from a more complex model (the teacher model) improves the convergence speed of the student model (which is our neural network) in cases where the number of sensor channels is greater than 10. However, we did not observe performance improvement. Hence, here the utilization of the knowledge distillation has the **only** purpose of accelerating the convergence of our neural network. Faster convergence speed of the student model with the use of distillation has been mathematically demonstrated in [21].

$$y'_k = \frac{\exp(l'_c/\tau)}{\sum_c \exp(l'_c/\tau)}, \tag{6}$$

Algorithm 1: Training of our activity classifier.

```
Create the neural network and define learning rate (e.g. 2e-3)
Load training data D_TRAIN and a pre-trained teacher model
Define the constants λ_C, λ_E, λ_CL, λ_D, E, τ and α
Set contains_unnecessary_channels = True
while contains_unnecessary_channels do
    for e = 1, e ≤ E, e++ do
        for all x_train in D_TRAIN do
            Compute gradients of the loss function L (Eq. 5)
            Update the weights of the neural network
        end
        Find the sets 𝕊^(1), ..., 𝕊^(N)
        Run validation to find F^(1), ..., F^(N), F = Σ_n^N F^(n)
        Save the model in memory if it has the highest F
    end
    Load the model that achieved the highest F
    Load F^(1), ..., F^(N) and 𝕊^(1), ..., 𝕊^(N)
    Set num_excluded_channels = 0
    for K = 1, K ≤ N, K++ do
        if F^(K) − F^(K−1) <α
            Exclude the sensor channel in 𝕊^(K) \𝕊^(K−1)
            num_excluded_channels += 1
    end
    if num_excluded_channels == 0
        Set contains_unnecessary_channels = False
        Measure the F1-score (Eq. 12) on the test set
    else
        Set N = N - num_excluded_channels
        Remove the inception blocks of excluded channels
        Reduce the learning rate (e.g., to 5e-4)
end
Return the trained classifier and a ranked list of
  non-excluded channels according to their v_i values.
```

where l'_c is the output of the classification layer of the teacher model for class c, and τ is termed as the softmax temperature and serves to smoothen the labels y'_k used in the student model. τ is a tuneable parameter usually set as $\tau \gg 1$. The performance of the student model is negatively affected as τ moves away from its optimal value in both directions.

3.5 Sensor channel selection

Before proceeding, we will explain the notation that will be used later. We define $\mathbb{S}$ as the set of all sensor channels (Eq. 7) that are being utilized in the training. The set $\mathbb{V}$ (Eq. 8) is defined as containing the values v_i (defined previously in Eq. 4) for all these sensor channels.

$$\mathbb{S} = \{s_i \mid i \in \mathbb{N}, 1 \leq i \leq N\}, \tag{7}$$

where N is the number of sensor channels in the training and $\mathbb{N}$ is the set of natural numbers.

$$\mathbb{V} = \{v_i \mid s_i \in \mathbb{S}\} \tag{8}$$

V_M is equal to the greatest real number m such that the set composed of all v_i that are greater than m has cardinality M. This is expressed in Eq. 9. In other words, V_M is the M-th highest value in $\mathbb{V}$.

$$V_M = \inf\{m \mid \text{card}(\{v \in \mathbb{V} \mid v > m\}) = M\} \tag{9}$$

The set $\mathbb{V}^{(M)}$ contains all values v_i that are greater or equal to V_M. That is, $\mathbb{V}^{(M)}$ contains the highest M values of $\mathbb{V}$. $\mathbb{S}^{(M)}$ is the set of all sensor channels whose values v_i are elements of $\mathbb{V}^{(M)}$. That is, all the M sensor channels of highest values v_i. Eq. 10 and Eq. 11 define these sets formally.

$$\mathbb{V}^{(M)} = \{v \in \mathbb{V} \mid v \geq V_M\} \tag{10}$$

$$\mathbb{S}^{(M)} = \{s_i \mid v_i \in \mathbb{V}^{(M)}\} \tag{11}$$

A pseudo-code of the entire training process is detailed in Algorithm 1. At the end of each epoch, as described in Algorithm 1, the validation is run for N times. The first validation is performed using $\mathbb{S}^{(1)}$ as input resulting in an F1-score (defined in Section 4.3) denoted as $F^{(1)}$. The second validation uses $\mathbb{S}^{(2)}$ and provides $F^{(2)}$, and so on until $\mathbb{S}^{(N)}$ and $F^{(N)}$. We define as F as $\sum_n^N F^{(n)}$. At the end of every E epochs, the model that achieved the highest F is loaded back into memory. The sensor channel selection then occurs as follows.

First, all the sets $\mathbb{S}^{(K)}\ \forall\ 1 < M \leq N, K \in \mathbb{N}$ are found. After this, the sensor channel s_i present in the set $\mathbb{S}^{(K)} \backslash \mathbb{S}^{(K-1)}$ is excluded if $F^{(K)} - F^{(K-1)} < \alpha \quad \forall\ 1 < K \leq N, K \in \mathbb{N}$, where α is a small real-valued positive constant (or zero) defined by the human prior to the training. In other words, a sensor channel s_i is excluded if its addition resulting in $\mathbb{S}^{(K-1)}$ becoming $\mathbb{S}^{(K)}$ does not improve the F1-score $F^{(K)}$ beyond a pre-defined threshold α. Note that, with this, the sole sensor channel contained in $\mathbb{S}^{(1)}$ is never excluded. The goal of the threshold α is to set a boundary between a negligible and an appreciable contribution to the performance. Sensor channels that do not provide at least a performance improvement of α are excluded as they are considered either redundant or irrelevant.

If no sensor channel is excluded, the training is complete. Otherwise, the training proceeds without the excluded sensor channels for another E epochs. Also, N is reduced to be equal to the number of sensor channels still present in the training and the sets $\mathbb{S}$ and $\mathbb{V}$ are redefined considering only the sensor channels still in use.

We propose to reduce the learning rate after removing some sensor channels. The reason for this is that we have found that the LSTM and classification layers overfit the data contained in the reduced number of sensor channels since their complexity becomes smaller than the required complexity for the new (and reduced) set of sensor channels.

At the end of the training, our algorithm provides a ranked list of the remained sensor channels according to their values v_i. Channels with a higher v_i are considered to provide more contribution to the classification task. At run-time, all the selected channels can be used for inference. However, when resources are more constrained, a subset of these channels containing the k most discriminative sensor channels can be used instead, with minor performance degradation. Naturally, k depends on the available resources at run-time. Before run-time, we calculate using the validation set the resource utilization for different values of k. These statistics are stored in a

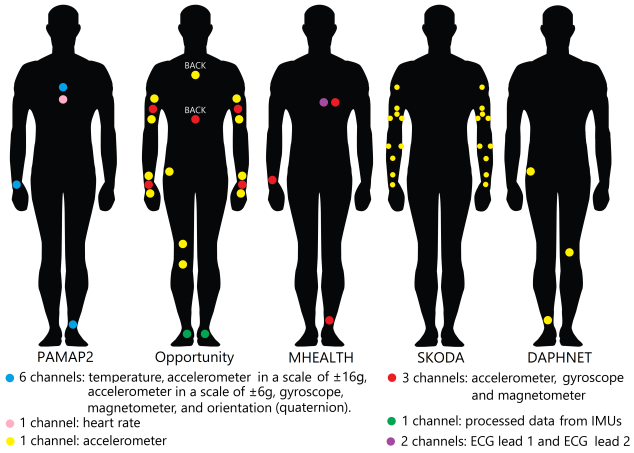

Figure 4: The sensor channels included in the test datasets. Best seen in color.

simple look-up table and loaded at run-time to select the value k that best fits within the computational budget available.

Network trimming. The inception blocks corresponding to the excluded sensor channels can be removed from the network. The performance drop associated with this trimming is negligible for two reasons: 1) due to the dropout layer, the network has learned to work with multiple combinations of sensor channels, and 2) the continuation of training with reduced learning rate better adapts the network to the absence of the excluded sensor channels. Also, at run-time, when resources are limited, the inception blocks that process the data from less discriminative selected channels can be loaded off the memory.

4 EXPERIMENTAL SETUP

4.1 Datasets

We used 5 public datasets for evaluating our method, each of which is described below. Fig. 4 illustrates the locations of the sensor channels in each dataset.

PAMAP2. The PAMAP2 dataset [22] encompasses 18 different physical activities. However, 6 of them are rarely present in the data. Following previous works [10, 24, 25], to avoid a heavily imbalanced training, only the remaining 12 activities are used in this work: lying quietly, sitting, standing, ironing, vacuum cleaning, ascending stairs, descending stairs, walking, Nordic walking, bicycling, running, and rope jumping. Transient activities (denoted as the null class) are discarded.

In total, 9 participants were designated for the data collection. They wore a heart rate monitor and 3 IMUs attached to the chest, hand, and ankle, respectively. Each IMU provided 6 different sensor channels. Hence, in total, 19 sensor channels. The sampling rate - originally at 100Hz - was reduced by decimation to 33.3Hz since faster sampling rates don't improve the performance for this dataset, but increase computation and memory utilization [24].

Opportunity. This dataset [6] incorporates 18 kitchen-related activities: null class, cleaning a table, opening/closing the fridge, opening/closing the dishwasher, opening/closing 3 different drawers (at different heights), opening/closing 2 different doors, toggling lights on and off, and drinking from a cup. The data were collected at 33 Hz from 4 participants equipped with sensors. In our experiments, only the sensory readings from the upper limbs, back, and both feet were considered (as in [10, 24, 25]). These readings come from 29 different sensor channels (Fig. 4).

Skoda. The Skoda dataset [32] covers 10 activities involved in a car maintenance scenario: writing down notes, opening/closing the engine hood, checking door gaps, opening/closing door, opening/closing two doors, checking the trunk gap, opening/closing the trunk, and checking the steering wheel. Only one subject took part in the data collection, wearing 10 3D-accelerometers on each arm - 20 sensor channels in total (Fig. 4). In our experiments, the sampling rate was decimated from 100Hz to 33Hz, for the same reason as in PAMAP2.

MHEALTH. The MHEALTH dataset [3] comprises 12 physical activities of varying intensities performed by 10 volunteers: standing still, sitting and relaxing, lying down, walking, climbing stairs, bending waist forward, elevating arms frontally, bending knees, cycling, jogging, running, and jumping to the front & back. IMUs were placed on the subject's chest, right wrist, and left ankle. Additionally, vital signs were recorded with 2-lead electrocardiogram (ECG) measurements. These readings sum up to 9 sensor channels (Fig. 4). The recordings were made at a 50Hz sampling rate.

Daphnet. This dataset [2] aims at detecting the freezing of gait in Parkinson's disease patients. It uses 3 wearable accelerometers (in total 3 sensor channels) on the ankle, thigh, and trunk, respectively. In total 10 subjects are included. The data contains 3 different labels: transient activities (discarded here), freezing of gait, and normal movements. In accordance with [33], we downsampled the data by decimation from 64Hz to 32Hz.

4.2 Data pre-processing

Prior to the training of the activity classifiers, we normalized all the data to zero mean and unit variance. Each dataset was segmented with sliding windows. Each window was labeled with the activity which lasts for the longest within the window. For the PAMAP2, following [11, 19, 24, 25, 33], the window size contained approximately 5.12 seconds with 78% overlap. These numbers are 1 and 50% for Skoda, Daphnet, MHEALTH, and Opportunity (also following [11, 24, 25]).

The proportions of the training, validation, and test data for Skoda and MHEALTH were chosen to be 0.70, 0.15, and 0.15, respectively. As for PAMAP2, we utilize the data from subject 5 for the validation set, subject 6 for the testing set, and the remaining subjects for the training set. This protocol is followed in diverse works [10, 11, 25, 28]. Also following [10, 11, 25, 28], for the Opportunity dataset, the validation set is composed of run 2 from subject 1, and the test set consists of runs 4 and 5 from subjects 2 and 3. The remaining data is assigned to the training set. Finally, for Daphnet, run 1 from subject 9 composed the validation set, whereas runs 1 and 2 from subject 2 formed the test set, with the remaining data for training. This is also followed in [11].

4.3 Evaluation metrics

We utilized the F1-score as the performance metric (Eq. 12). The F1-score can account for class imbalance in the dataset [19], which is a common case in HAR.

$$F_1 = \frac{2TP_i}{2TP_i + FP_i + FN_i}, \tag{12}$$

where TP_i, FP_i, and FN_i represent the number of true positives, false positives, and false negatives of a certain class, respectively. To obtain a single measure for all classes, we calculate a weighted average of the F1-score for all classes, weighting each class according to the number of samples available for it.

Other evaluation metrics include the memory footprint of the neural network, the inference time of a prediction, and the energy consumption per prediction. These metrics were measured on the Raspberry Pi 4B microcontroller - a low-cost device for general purposes - which is powered with a quad-core Cortex-A72 64-bit System on a Chip.

4.4 The code

The training of the pipeline was implemented in Python 3.8.5 with TensorFlow 2.3.0 and executed on an Intel Xeon Gold 6134 at 3.20GHz and an NVIDIA Tesla V100 16GB of RAM. The inference, however, took place on a Raspberry Pi 4B - as mentioned in Sec 4.3. Adam was used as the optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The mini-batch size was set to 32. The LSTM layer included 32 hidden units. The dropout rate was set to 0.35. The choice of these hyper-parameters was made by trial and error utilizing the validation sets. Our neural network is stable in the sense that different combinations of hyper-parameters lead to very similar results.

We used InnoHAR [28] as the teacher model for the knowledge distillation and for later comparisons in Sec. 5, with the same hyper-parameters provided in the authors' paper [28]. As mentioned in Sec. 3.4, knowledge distillation is only utilized to improve the convergence speed at the beginning of the training for cases where the number of channels is above 10 - i.e. in the PAMAP2, Skoda, and Opportunity datasets. Distillation for MHEALTH and Daphnet has proved to be ineffective because the convergence speed for them is already fairly fast due to the small number of channels.

The threshold α was set to 5e-3. This signifies that we consider improvements in F1-score lower than 5e-3 to be negligible and channels whose contribution is within this interval should be removed. The number of epochs E is 50. The learning rate was set to 2e-3 and reduced to 5e-4 after the first 50 epochs. The constants λ_C, λ_E and λ_D were set - respectively - to 1, 0.4 and 0.5. We utilized a softmax temperature $\tau = 10$.

4.5 Comparison with previous works

We implemented Heuristic-based Assessment (HQA) [17], Practical Body Networking (PBN) [15], Minimal Redundancy and Maximal Relevancy Ensemble Pruning (mRMR EP) [5] and Naïve Bayes Fusion with Exhaustive Search [30].

HQA [17] was originally devised for device selection - where each device consists of a group of sensor channels. Here, however, we consider that each device consists of a sole sensor channel. Also, for a fairer comparison, the hyper-parameters for the methods PBN [15], mRMR EP [5] and NBF + ES [30] were chosen so that the resulting number of selected channels were equal to that of our method. In NBF + ES, we limited the exhaustive search to 50 random combinations of channels, since analyzing all the possible combinations would result in days of uninterrupted code execution. Choosing the number of channels for HQA [17] is not possible since - at every k seconds - this method selects one device among all possible ones and only utilizes it to generate the predictions until a new device is chosen in the following selection. As proposed in the original paper, We set k equal to 10 times the length of the sliding window.

Finally, as discussed in Section 2.2, all four channel selection methods used for comparison utilize an ensemble of classifiers where each classifier is responsible for generating predictions to a sensor channel and the channel selection problem is transformed into a classifier selection problem. The classifiers in the ensemble were chosen to be a simple version of InnoHAR with 12 filters on each convolutional layer - instead of the original 128 filters. This number of filters was found - by trial and error using the validation sets of each dataset - to be the minimum one that provides in average best F1-score across all datasets.

Dataset: MHEALTH
Iteration 1: F1-score of 0.982, ECG monitors and gyroscopes removed. Only accelerometers and gyroscopes remaining.
Iteration 2: F1-score of 0.983, right arm acc. removed. Remain left ankle acc., chest acc., right arm mag. and left ankle mag.
Iteration 3: F1-score of 0.992. No channels removed. The remaining channels are the same as in the previous iteration.

Table 2: Demonstration of the pipeline's results through all iterations for MHEALTH.

Dataset	Selected channels by our method
Daphnet	**2 out of 3:** 1. Trunk acc. and 2. Shank acc.
MHEALTH	**4 out of 9:** 1. Left ankle acc., 2. Chest acc., 3. Right arm acc., and 4. Left ankle mag.
PAMAP2	**5 out of 19:** 1. Ankle mag., 2. Chest ±16g acc., 3. Hand ±16g acc., 4. Ankle ±6g acc., and 5. Chest mag.
Opportunity	**4 out of 29:** 1. Right wrist mag., 2. Center of the back mag., 3. Right biceps acc., and 4. Left wrist acc.
SKODA	**8 out of 20 (acc. only):** 1. Slightly above left wrist, 2. Slightly below right wrist, 3. Left-side of upper right arm, 4. Right-side of upper right arm, 5. Left arm brachoradialis muscle, 6. Left arm deltoid muscle, 7. Right arm deltoid muscle, and 8. Right-side of the upper left arm

Table 3: The sensor channel selection - in order from most to least discriminative - provided by our algorithm. Abbreviations: acc. for accelerometer, gyro. for gyroscope and mag. for magnetometer.

5 EXPERIMENTS

In this section, we first demonstrate the results at all stages of our pipeline for MHEALTH (Sec. 5.1). Next, in Sec. 5.2, we show the final results - in terms of the evaluation metrics described in Sec. 4.3 - for all datasets and compare them to related works in sensor channel selection [5, 15, 17, 30], as well as to InnoHAR [28] - which does not perform sensor channel selection. We then evaluate the trade-off between the number of sensor channels and classification performance. In Sec. 5.3 we discuss the criteria for

Method	F1-score, memory footprint (MB), inference time (ms) and energy consumption per prediction (mJ)																			
	Daphnet				MHEALTH				PAMAP2				Opportunity				Skoda			
InnoHAR [28]	0.889	17.1	28.7	79.2	**1.000**	23.9	48.4	140.4	**0.916**	70.9	182.1	540.0	**0.898**	17.9	33.0	104.4	**0.979**	17.1	26.8	80.8
Ours	**0.901**	**2.0**	**0.4**	**1.8**	0.990	**2.9**	**1.1**	**3.6**	0.884	**3.5**	**4.0**	**12.9**	0.864	**1.9**	**0.8**	**2.1**	0.926	**3.0**	**1.5**	**5.0**
HQA [17]	0.838	5.8	0.6	2.3	0.892	27.9	1.76	5.3	0.776	59.8	9.97	30.2	0.782	85.5	3.4	10.9	0.885	38.2	1.9	5.9
PBN [15]	0.855				0.993				0.875				0.804				0.925			
mRMR EP [5]	0.841	3.8	1.1	3.9	0.967	12.4	3.9	11.8	0.856	16.0	17.8	54.0	0.760	11.8	3.6	11.5	0.913	15.2	5.2	16.2
NBF + ES [30]	0.856				0.990				0.820				0.778				0.927			

Table 4: Comparison between our method, InnoHAR [28] and related works in sensor channel selection [5, 15, 17, 30]. InnoHAR uses all possible channels and is included here to provide contrast with selection methods. At inference time, the methods PBN [15], mRMR EP [5] and NBF + ES [30] only differ with respect to the fusion of the predictions of each classifier. The resource utilization involved in this fusion stage is negligible for all three methods, hence the identical values in resource utilization.

Datasets	PAM.	MHE.	Skoda	Daph.	Opp.
Number of channels	2	1	2	1	1
F1-score	0.707	0.779	0.679	0.855	0.737
Memory footprint (MB)	3.14	1.87	1.87	1.91	1.91
Inference time (ms)	1.79	0.36	0.44	0.26	0.25
Energy cons. per pred. (mJ)	5.04	1.08	1.44	0.68	0.75
Number of channels	3	2	4		2
F1-score	0.780	0.960	0.811		0.811
Memory footprint (MB)	3.29	1.95	1.94	-	1.92
Inference time (ms)	2.52	0.63	0.80		0.44
Energy cons. per pred. (mJ)	7.92	1.80	2.51		1.08
Number of channels	4	3	6		3
F1-score	0.865	0.970	0.913		0.825
Memory footprint (MB)	3.31	2.00	2.05	-	2.01
Inference time (ms)	3.28	0.90	1.16		0.62
Energy cons. per pred. (mJ)	10.08	2.52	3.24		1.80

Table 5: Performance and inference expenses of our neural network for different numbers of sensor channels. The Daphnet dataset featured only 2 selected channels. Therefore, only one - instead of 3 - configuration was tested here. Abbreviations: PAM. for PAMAP2, Daph. for Daphnet, MHEA. for MHEALTH, and Opp. for Opportunity.

optimal sensor channel selection observed from our experiments. Finally, we provide ablation studies in Sec. 5.4.

5.1 Demonstrating the pipeline

A short demonstration - for the MHEALTH dataset - of all stages of our pipeline is outlined in Table 2. In total, 3 iterations were needed. The process started with 9 sensor channels and, at the end of each iteration, a reduced number of them remained. However, the last iteration left the sensor channels unchanged and, as a consequence, it marked the end of the pipeline. The performance increases after each iteration, despite sensor channels being excluded. The reason is two-fold: 1) the excluded sensor channels contain irrelevancies and unnecessary redundancies, and 2) the classifier is being continuously trained with a reduced learning rate.

5.2 Comparison with the state-of-the-art

Our sensor channel selection algorithm provides a ranked list of sensor channels according to their contribution to the overall performance - Table 3. In the listing, the first channel is the most discriminative one (i.e. highest v_i according to Eq. 4), and vice-versa for the last channel. In scenarios of sufficient computational resources, all the selected channels can be used to obtain the best performance possible. However, in cases of limited resources, a subset of the selected channels containing those that contribute the most can instead be used. The discussion on the selection for each dataset is included in Section 5.3.

Table 4 presents the results of our neural network (trained assisted with the sensor channel selection) compared with other selection methods [5, 15, 17, 30] and InnoHAR [28] - a DL model for HAR that delivers state-of-the-art performance and utilizes all the available sensor channels. In terms of prediction performance, our neural network was able to deliver F1-scores closer - or even higher as for the Daphnet dataset - to the case where all sensor channels are utilized (in the InnoHAR [28] row). This is due to the characteristic of our neural network of learning inter-channel dependencies. This characteristic is not present in the works we compare with since they train separate and independent classifiers for each sensor channel, which leads to a degradation in prediction performance for datasets where considering intricate relationships between the sensor channels may be determinant to more accurate predictions - as it is the case for Daphnet, PAMAP2, and Opportunity.

Comparing resource utilization with other selection methods, we observe that our neural network occupies from 76% to 93% less memory - averaged across all datasets. Improvements are also observed in inference time, where we achieve from 45% to 75% faster prediction delay on average. Finally, the energy consumption is on average from 56% to 76% smaller for our neural network. Compared to InnoHAR [28], our neural network achieves at least 91% of savings in resource utilization in any of the considered metrics.

This considerable leap in resource efficiency is attributed to two characteristics of our neural network's architecture. First, in our method, the fusion of the sensor channels happens inside the neural network and before the recurrent and classification layers. This signifies that the same recurrent and classification layers are utilized for all sensor channels. In other selection methods, each sensor channel has its own classifier with individual recurrent and classification layers, which adds to resource utilization. The second reason is that our neural network relies more on inter-channel than intra-channel dependencies to generate a prediction; which allows for more light-weight layers that learn intra-channel dependencies. This is not the case with other selection methods since each channel has its own classifier that is trained independently of the others, thus relying mostly on intra-channel dependencies and consequently requiring more complex structures in each classifier.

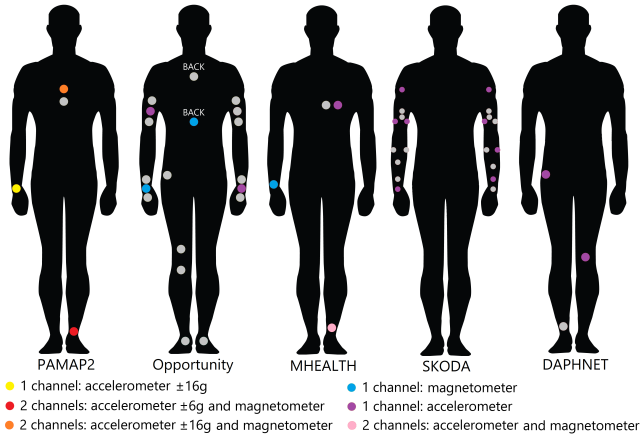

Figure 5: The selected sensor channels of the 5 public datasets. Excluded sensors are shown in grey color.

For each dataset, we also tested classification models with different numbers of the most discriminative sensor channels as input. The results are presented in Table 5. As the number of channels is reduced, there is a considerable reduction in inference time and energy consumption with only a reasonable performance degradation. The memory footprint, however, is hardly affected by the reduction of sensor channels. We are unsure as to why this is the case, but we conjecture that larger savings in memory footprint when reducing the number of channels are possible by tweaking the deployment of the neural network to the microcontroller.

We performed these tests to demonstrate the ability of our neural network when working - without major performance drops - with a smaller set of the channels it was trained with, in order to meet a more limited resource availability. As described in Sec. 3.5, at run-time, statistics of resource utilization as those in Table 5 can be utilized to define the number of k most discriminative sensor channels to adapt to the resource utilization constraints. For instance, for the PAMAP2 dataset, utilizing only the two most discriminative channels results in an F1-score of 0.707 while reducing the inference time and energy consumption to approximately 60% of the original value (when all the 5 selected sensor channels are used). Replacing with zeros all sensor channels, except for the two most discriminative ones, in InnoHAR, results in a performance drop from 0.916 to 0.229. This major performance drop is not only exclusive to InnoHAR. The training of standard DL models for HAR leads to built co-adaptations between sensor channels. As discussed previously in Sec. 3.1, we approached this issue by employing random dropouts of sensor channels during the training.

The comparison is not performed with other selection methods for the following reasons. HQA [17] strictly uses only one sensor channel to generate a prediction. PBN [15], mRMR EP [5] and NBF + ES [30] can dynamically change the number of used channels at run-time, however, these methods do not provide a ranked list of channels according to their contribution to the classification.

5.3 Criteria of sensor channel selection

Our experimental results prove the effectiveness of our sensor channel selection method and also provide insights on the following characteristics of HAR and criteria of sensor channel selection. Figure 5 illustrates the results of sensor channel selection for all datasets.

5.3.1 Redundancy removal. It is intuitive that sensors located extremely close to each other share redundancies since we can expect that they capture the same information. This is the case of the Opportunity and Skoda datasets. In the case of Skoda, there are two closely-spaced accelerometers on the wrist. Our method selects only one of them. The clutter located on the lower and upper arm is also minimized via the selection. In the case of Opportunity, three sensors are placed on the wrist, from which only one is selected. In general, there is a maximization of the distance between sensors for both datasets.

5.3.2 Irrelevancy removal. The evidence favoring the characteristic of irrelevancy removal of our method is seen in PAMAP2, MHEALTH, Opportunity, and Daphnet. One of the first things observed for the PAMAP2 dataset is that the heart rate sensor channel was excluded. The activities in the dataset can be classified into 1) activities of low intensity (lying down, sitting, standing), activities of medium intensity (ironing, vacuum cleaning, ascending and descending stairs, walking, and Nordic walking), and activities of high intensity (running, rope jumping and bicycling). The information about the heart rate is only able to classify among these groups of activities. The remaining sensor channels, however, are able to give more precise information about the activity. For example, in high-intensity activities, the accelerometer placed on the hand can provide information about whether or not the participant is rope jumping. The accelerometer and magnetometer on the ankle can then classify between running or bicycling. Therefore, it is intuitive that the heart rate can be excluded.

PAMAP2 also contains sensor channels with temperature information. These channels, however, are not present in the selection. A study performed in [18] showed that the skin temperature of a person increases in response to high-intensity physical activities. However, the variations in temperature can persist even after 24 hours after the physical activity. Therefore, using the skin temperature to detect activities (or even the intensity level of the activities) may not be appropriate. Our sensor channel selection method automatically excluded these channels. Finally, as described in the documentation of the PAMAP2 (available on the UCI Machine Learning Repository online), the orientation sensor channels are invalid for the data collection (the reason is not specified). These channels have also been opted out in the selection.

In MHEALTH, the information about the heart activity - from ECG lead 1 and lead 2 - was disregarded in the selection. The same explanation used for the PAMAP2 holds true here since the activities and sensor channels involved in these two datasets are similar. Concerning Opportunity, the set of activities present in the dataset are activities in which the upper part of the body is the main actor. Our method was able to understand this by excluding all sensor channels from the lower part of the body.

In the last case of irrelevancy removal, in the Daphnet dataset, the sensor at the right leg's ankle was excluded from the selection. When the freezing of gait occurs, the movement of the patient's feet suddenly stops, while the torso still carries forward momentum. Since the feet cannot move, the momentum on the upper body is

directed downwards, which causes falls in some cases. Therefore, it is expected that sensors placed at the height of the ankle may not detect the change in direction of the momentum of the upper body. This explains the absence of the ankle's accelerometer in the selection.

5.3.3 Ranking the most discriminative channels higher. The effectiveness of our method in ranking the most discriminative sensor channels on the top is observed in all datasets. In PAMAP2, regarding the scale of the accelerometers, ±16g seems to be preferred. This can be explained by the fact that, in some activities, the ±6g variant can get quickly saturated.

In Opportunity, at least 7 of the activities involve leaning the back forward to reach objects at different heights. This explains the magnetometer channel ranking at the top since it is able to provide information on the orientation of the participant's back. We hypothesize that the participants used the right side of the body predominantly. This hypothesis can explain why the right arm sensor channels are, in general, placed above that of the left arm.

In MHEALTH, the most discriminative channels were listed as the ankle's and chest's accelerometers. This combination is effective since 9 out of the 10 activities - that is, excluding the activity of elevating the arms - can be well differentiated by patterns of movements from the chest and the ankle. Less discriminative is the right arm's magnetometer since it is mostly useful for only one activity: elevating the arms. We hypothesize that the ankle's magnetometer is useful to differentiate between sitting and standing up since the ankle's orientation can differ in these activities.

As for Skoda, the sensors placed on the wrist ranked at the top since the amplitude of the movement there is higher, which can be more helpful in discriminating between activities. For the same reason, sensors ranked at the bottom are those located on the upper arm, for instance, on the deltoid muscle. Finally, for Daphnet, the accelerometer on the trunk is selected to be the most discriminative. As we discussed previously, it is near the trunk where the change in the direction of the momentum occurs, thus sensors placed at that height are expected to be more discriminative.

5.3.4 Symmetry. When activities are equally performed by both sides of the body and when the sensor placement also exhibits symmetry with respect to the center-line of the body, the selection of optimal sensor channels is expected to also follow the symmetry. This is observed in Skoda and Opportunity. However, in Opportunity, the symmetry is less pronounced due to the presence of an additional channel placed on the right arm. This can be explained by the same hypothesis as used before: the participants in the training set use the right hand predominantly.

5.3.5 Consistency. In the MHEALTH dataset, it is noticeable that the set of selected sensor channels shows similarity to that of the PAMAP2 dataset. This shows evidence for consistency in the results of our method since both datasets encompass nearly identical activities, sensor placement, and sensor type (IMUs and heart activity sensors). This also demonstrates that prior knowledge from prior sensor selection cases can serve as concrete guidance to other cases. For instance, sensors placed on the ankle should provide an advantage in activities running, jogging, walking, climbing stairs, jumping, and cycling. Also, a heart rate monitor is irrelevant for the same type of activities.

Ablation	**Without dropout**		**Without entropy term**	
Dataset	**PAMAP2**	**Opportunity**	**PAMAP2**	**Opportunity**
F1-score	0.757	0.858	0.886	0.870
# of selected channels	6	11	8	7

Table 6: Results of ablation studies in the absence of the dropout layer and the entropy term in the loss function.

5.4 Ablation studies

In this section, we propose to remove the dropout layer and the entropy term in the loss function, respectively, to evaluate their impact on the results. We limited our attention to the two largest and hardest datasets - PAMAP2 and Opportunity. Table 6 presents the results.

Removing the dropout degrades the performance especially in the PAMAP2 case where the degradation reaches roughly 13%. When the neural network is trained without dropout, it learns to rely on features that include built co-adaptations between sensor channels. For this reason, by removing sensor channels, the performance can degrade unless the network is sufficiently trained without the removed channels. It is also noticed dropping out channels during training also contributes to a reduced selection, as it helps the network to better learn about the contribution of each channel.

In the absence of the entropy term, the variation in performance is negligible. However, the cost of removing this term is evidenced in the higher number of selected sensor channels. The reason for this is that the entropy term enforces the network to learn to create an imbalance of importance in the sensor channels while maintaining high performance. This imbalance results in the neural network learning which channels are redundant or irrelevant for the classification. The entropy term is, therefore, essential in our design.

6 DISCUSSION

We discuss in this section the advantages and limitations of our sensor channel selection method, as well as a possible research direction for future work.

6.1 Advantages

Our proposed solution advances the previous works through the following designs: resource efficiency, insights on the black-box DL models regarding sensor channels, adaptability to variable resource constraints, and sensor malfunction robustness.

Lightweight. Sensor channels that were not selected are simply removed from the input of the network, instead of being replaced with zeros. This means that computational power can be spared, especially in cases where sparse tensor algebra is unavailable or not used - which is often the case of common off-the-shelf microcontrollers. Removing sensor channels from the input also makes the network slimmer, since the weights responsible for modeling dependencies in the excluded channels are no longer needed.

Insightful. Our pipeline provides insights - through the ranking of the v_i values - on the contribution of each sensor channel to the classification task, which to a certain extent helps to unveil the black-box nature of DL models.

Datasets	PAMAP2	MHEALTH	Skoda	Daphnet	Opp.
# of epochs	250	150	150	50	300
Duration (h)	5	1.5	3	< 2.5	11

Table 7: The number of epochs and the approximate training duration for each dataset. Abbreviation: Opp. for Opportunity.

Modularity. The number of sensor channels used at run-time can vary. In cases when the computational resources should be spared, the number can be reduced without retraining the classifier, taking into account the performance requirements.

Robustness. The use of the dropout layer forces the network to learn to work with random combinations of sensor channels, thus giving it the ability to deal with sensor malfunction more robustly in comparison to other networks.

6.2 Limitations

The limitations are related to the long training duration and to the fact that it does not rule out the need for data collection with several sensors.

Training duration. The presence of the dropout layer increases the convergence time of the learning since multiple variations of the same network are being trained [26]. Fusing the channels by performing average pooling across channels is very similar to a global average pooling layer, which also slows the convergence speed. The use of knowledge distillation reduces the effect of slow convergence. However, it does not completely solve it. Table 7 provides the number of epochs and the approximate duration of the training for each dataset. We estimate that the total duration spent on training the classifier is 4 - 5 times longer than InnoHAR, due to the repeated iterations for sensor channel selection. However, compared to exhaustive search, this is many orders of magnitude faster than training all possible sensor channels combinations (e.g., more than 500 million combinations in the case of Opportunity). Also, based on our experiments, the training of our neural network is also up to 2 times faster than training an ensemble of weak classifiers [5, 15, 17, 30].

Adaptability to dynamic context. Our selection algorithm considers a static context where the importance of the sensors doesn't vary according to the activity or the subject who performs the activities. However, this can be addressed by making the trainable weights $w_{i|j}$ dependent on the activity and subject. In Algorithm 1, this signifies that the sets $\mathbb{S}$ and the values $F^{(\cdot)}$ also depend on the activity or subject and F becomes the average of all the values $F^{(\cdot)}$ across all activities or subjects. The ranked list of channels then becomes also dependent on the activity or subject.

Physical prototypes. Our method still does not remove the need for creating physical prototypes with multiple sensors and for collecting data. For future work, we would like to integrate our method into a data collection simulator that can generate synthetic data from artificial sensors, thus resulting in more significant time and cost savings involved in developing prototypes for HAR systems.

7 CONCLUSIONS

We have proposed a sensor channel selection pipeline composed of a modular neural network that learns to minimize the use of sensor channels by understanding the redundant and irrelevant relations between them. Also contained in the pipeline, our sensor channel selection algorithm extracts the knowledge of the network regarding the channels, selects the most discriminative ones, and discards those that provide a negligible contribution to the overall performance. The information on the contribution of each sensor channel is easily accessible which provides insights that help to reduce the black-box nature of DL models for HAR.

Compared to recent work, our selection algorithm led to significantly higher classification performance and reduced resource utilization, as well as showed consistency regardless of the number of sensor channels in the dataset. In addition, we provided and discussed evidence in favor of the claimed features of our pipeline, including its characteristic of being able to work with a subset of the sensor channels used during training with minor performance degradation. Finally, even though our experiments focused on HAR, we believe that our pipeline can be applied to other multi-input problems in DL with little or no modification, due to the fact that all methods in the pipeline are general.

ACKNOWLEDGEMENT

This work was funded by Business Finland (grant number: 1573/31/2020).

REFERENCES

[1] O. Aziz, S. N. Robinovitch, and E. J. Park. 2016. Identifying the number and location of body worn sensors to accurately classify walking, transferring and sedentary activities. In *2016 EMBC*. 5003–5006.
[2] M. Bachlin, M. Plotnik, D. Roggen, I. Maidan, J. M. Hausdorff, N. Giladi, and G. Troster. 2010. Wearable Assistant for Parkinson's Disease Patients With the Freezing of Gait Symptom. *IEEE Transactions on Information Technology in Biomedicine* 14, 2 (March 2010), 436–446.
[3] O. Banos, J. A. Moral-Munoz, I. Diaz-Reyes, M. Arroyo-Morales, M. Damas, E. Herrera-Viedma, C. S. Hong, S. Lee, H. Pomares, I. Rojas, and C. Villalonga. 2015. mDurance: A Novel Mobile Health System to Support Trunk Endurance Assessment. *Sensors* 15, 6 (2015), 13159–13183.
[4] K. A. Bhaskaran, A. G. Nair, K. D. Ram, K. Ananthanarayanan, and H. R. Nandi Vardhan. 2016. Smart gloves for hand gesture recognition: Sign language to speech conversion system. In *2016 RAHA*. 1–6.
[5] J. Cao, W. Li, C. Ma, and Z. Tao. 2018. Optimizing Multi-Sensor Deployment via Ensemble Pruning for Wearable Activity Recognition. *Inf. Fusion* 41, C (May 2018), 68–79.
[6] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. del R. Millán, and D. Roggen. 2013. The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recognition Letters* 34, 15 (2013), 2033 – 2042.
[7] Y. Du, Y. Lim, and Y. Tan. 2019. A Novel Human Activity Recognition and Prediction in Smart Home Based on Interaction. *Sensors* 19, 20 (2019), 1–16.
[8] O. Ertugrul and Y. Kaya. 2016. Determining the optimal number of body-worn sensors for human activity recognition. *Soft Computing* 21 (02 2016), 5053–5060.
[9] M. Espinilla, J. Medina, A. Calzada, J. Liu, L. Martínez, and C. Nugent. 2017. Optimizing the configuration of an heterogeneous architecture of sensors for activity recognition, using the extended belief rule-based inference methodology. *Microprocessors and Microsystems* 52 (2017), 381 – 390.
[10] Y. Guan and T. Plötz. 2017. Ensembles of Deep LSTM Learners for Activity Recognition Using Wearables. *IMWUT* 1, 2, Article 11 (June 2017), 28 pages.
[11] N. Y. Hammerla, S. Halloran, and T. Plötz. 2016. Deep, Convolutional, and Recurrent Models for Human Activity Recognition Using Wearables. In *IJCAI'16*. AAAI Press, 1533–1540.
[12] G. Hinton, O. Vinyals, and J. Dean. 2015. Distilling the Knowledge in a Neural Network. *arXiv e-prints* arXiv:abs/1503.02531 (March 2015), 1–9.
[13] S. Hochreiter and J. Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780.
[14] J. Hsiao, Y. Deng, T. Pao, H. Chou, and J. Chang. 2017. Design of a Wireless 3D Hand Motion Tracking and Gesture Recognition Glove for Virtual Reality Applications, Vol. ASME 2017 Conference on Information Storage and Processing Systems. 1–3.
[15] M. Keally, G. Zhou, G. Xing, J. Wu, and A. Pyles. 2011. PBN: Towards Practical Activity Recognition Using Smartphone-Based Body Sensor Networks. In *Sensys*

'11 (Seattle, Washington). Association for Computing Machinery, New York, NY, USA, 246–259.

[16] J. Lei Ba, J. R. Kiros, and G. E. Hinton. 2016. Layer Normalization. *arXiv e-prints* arXiv:abs/1607.06450 (July 2016), 1–14.

[17] C. Min, A. Montanari, A. Mathur, and F. Kawsar. 2019. A Closer Look at Quality-Aware Runtime Assessment of Sensing Models in Multi-Device Environments. In *Sensys '19* (New York, New York). Association for Computing Machinery, New York, NY, USA, 271–284.

[18] E. B. Neves, J. Vilaça-Alves, N. Antunes, I. M. V. Felisberto, C. Rosa, and V. M. Reis. 2015. Different responses of the skin temperature to physical exercise: Systematic review. In *2015 EMBC*. 1307–1310.

[19] F. J. Ordóñez and D. Roggen. 2016. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* 16, 1 (2016), 1–25.

[20] L. Peng, L. Chen, Z. Ye, and Y. Zhang. 2018. AROMA: A Deep Multi-Task Learning Based Simple and Complex Human Activity Recognition Method Using Wearable Sensors. *ACM IMWUT* 2, 2, Article 74 (July 2018), 16 pages.

[21] M. Phuong and C. Lampert. 2019. Towards Understanding Knowledge Distillation. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, K. Chaudhuri and R. Salakhutdinov (Eds.). PMLR, Long Beach, California, USA, 5142–5151.

[22] A. Reiss and D. Stricker. 2012. Introducing a New Benchmarked Dataset for Activity Monitoring. In *2012 16th International Symposium on Wearable Computers*. 108–109.

[23] L. Schrader, A. Vargas Toro, S. Konietzny, S. Rüping, B. Schäpers, M. Steinböck, C. Krewer, F. Müller, J. Güttler, and T. Bock. 2020. Advanced Sensing and Human Activity Recognition in Early Intervention and Rehabilitation of Elderly People. *Journal of Population Ageing* 13, 2 (01 Jun 2020), 139–165.

[24] C. F. Souza Leite and Y. Xiao. 2020. Improving Cross-Subject Activity Recognition via Adversarial Learning. *IEEE Access* 8 (2020), 90542–90554.

[25] C. F. Souza Leite and Y. Xiao. 2020. Improving Resource Efficiency of Deep Activity Recognition via Redundancy Reduction. In *HotMobile '20*. Association for Computing Machinery, 33–38.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15, 56 (2014), 1929–1958.

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2014. Going Deeper with Convolutions. *CoRR* arXiV:abs/1409.4842 (2014), 1–12.

[28] C. Xu, D. Chai, J. He, X. Zhang, and S. Duan. 2019. InnoHAR: A Deep Neural Network for Complex Human Activity Recognition. *IEEE Access* 7 (2019), 9893–9902.

[29] X. Yang, Yiqiang Chen, H. Yu, Y. Zhang, W. Lu, and R. Sun. [n.d.]. Instance-wise Dynamic Sensor Selection for Human Activity Recognition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*. AAAI Press, 1104–1111.

[30] P. Zappi, C. Lombriser, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Tröster. 2008. Activity Recognition from On-Body Sensors: Accuracy-Power Trade-Off by Dynamic Sensor Selection. In *Wireless Sensor Networks*, Verdone R. (Ed.). Springer, Berlin, Heidelberg, 17–33.

[31] P. Zappi, D. Roggen, E. Farella, G. Tröster, and L. Benini. 2012. Network-Level Power-Performance Trade-Off in Wearable Activity Recognition: A Dynamic Sensor Selection Approach. *ACM Trans. Embed. Comput. Syst.* 11, 3, Article 68 (Sept. 2012), 30 pages.

[32] P. Zappi, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Troster. 2007. Activity recognition from on-body sensors by classifier fusion: sensor scalability and robustness. In *2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information*. 281–286.

[33] M. Zeng, H. Gao, T. Yu, O. J. Mengshoel, H; Langseth, I. Lane, and X. Liu. 2018. Understanding and Improving Recurrent Networks for Human Activity Recognition by Continuous Attention. In *ISWC '18* (Singapore, Singapore). ACM, 56–63.