Hi-OSCAR: Hierarchical Open-set Classifier for Human Activity Recognition

Conor McCarthy[1], Loes Quirijnen[2], Jan Peter van Zandwijk[2,3], Zeno Geradts[2,1], and Marcel Worring[1]

[1]University of Amsterdam, Amsterdam, The Netherlands
[2]Netherlands Forensic Institute (NFI), The Hague, The Netherlands
[3]Amsterdam University of Applied Sciences, Amsterdam, The Netherlands

Abstract

Within Human Activity Recognition (HAR), there is an insurmountable gap between the range of activities performed in life and those that can be captured in an annotated sensor dataset used in training. Failure to properly handle unseen activities seriously undermines any HAR classifier's reliability. Additionally within HAR, not all classes are equally dissimilar, some significantly overlap or encompass other sub-activities. Based on these observations, we arrange activity classes into a structured hierarchy. From there, we propose Hi-OSCAR: a **Hi**erarchical **O**pen-set **C**lassifier for **A**ctivity **R**ecognition, that can identify known activities at state-of-the-art accuracy while simultaneously rejecting unknown activities. This not only enables open-set classification, but also allows for unknown classes to be localized to the nearest internal node, providing insight beyond a binary "known/unknown" classification. To facilitate this and future open-set HAR research, we collected a new dataset: NFI_FARED. NFI_FARED contains data from multiple subjects performing nineteen activities from a range of contexts, including daily living, commuting, and rapid movements, which is fully public and available for download[1].

Introduction

Human Activity Recognition (HAR) is the process of classifying activities using data collected from one or more sensors placed either on the body or in the environment where the activity is taking place. With the continuous improvement of sensor technology, body-worn HAR has received a lot of attention because of its potential scalability, high privacy, and relatively low cost when compared to using cameras and other environment-based sensors. Body-worn HAR has found applications in sports [40, 39], education [28, 29], medicine [79, 19, 9], smart homes [4], and forensics [37, 76, 77]. Apart from the lower cost and wider selection of sensors being available now, the continued advancement of AI-based techniques has improved classification performance and added capabilities.

AI techniques for HAR rely on large annotated datasets. A limiting factor for practical applications is the coverage gap between datasets for training annotated with a limited number of activities and real-world activity variations. In any uncontrolled setting, the number of encountered activities will greatly outnumber the set of classes in the data used for training. Datasets are needed that are taking in real-world conditions and are annotated with a large variety of activities. But even when AI tools are trained with a large number of activities, there will always be activities that have not yet been encountered. Failure to appropriately flag and process such unknown activities can render the output of activity classification models worthless. Consequently, an area of HAR seeing increased attention is

[1]`https://osf.io/sczrh/`

open-set classification. Open-set classification models are able to label samples as Out-of-Distribution (OOD) if the class is from outside the training set. In contrast, closed-set classification restricts the model to only recognise In-Distribution (ID) classes present in the training set [20, 36]. A closed-set approach forces the model to erroneously classify OOD input as one of the classes from the training set, sometimes with high reported certainty. Given the wide array of activities that can reasonably be encountered in practical settings, open set classification is a necessary component of HAR.

While typical approaches keep activities organised in a flat structure without leveraging inter-class relations, structuring classes as a hierarchy can be a useful method for improving open-set performance [48]. Arranging the classes into a hierarchical tree structure groups similar classes close to each other, with narrowly-defined classes below their more generally-defined supersets. A hierarchy better represents HAR data, where not all activities are equally dissimilar, and some can often be subsets of the other. For example, *walking* and *walking upstairs* are more closely related than *walking* and *punching*. The hierarchy embeds additional information into predictions. Our model, Hi-OSCAR, then makes predictions by traversing the tree through multiple internal classification decisions (Figure 1). When encountering and rejecting an OOD sample, the traversed path can still be inspected, and provide insight into which known classes are similar to this OOD activity. Hi-OSCAR's open-set classification localises an OOD sample to the nearest internal node without changing the underlying hierarchy. For example, our model might localise the unknown class *sitting* to an internal node close to the known class *standing*. In the case of the unknown class being completely different to anything in the training set, nodes at or near the Root of the hierarchy would be returned. Such information is also insightful when interpreting the unknown activity. In our method, we apply an activity-level hierarchy to mirror the structure of real-life HAR classes.

It is important to create a hierarchy that is both semantically meaningful and aligns with the data signatures. Previous work in HAR considered a hierarchy of simple gestures or sub-activities which when performed in a sequence form a complete activity class [73, 80]. This type of hierarchy falls short of then linking together the higher level classes in a meaningful way. This has been done in other domains which borrow existing hierarchies based on domain knowledge, such as the ImageNet dataset adopting the Wordnet hierarchy [42]. No such hierarchy exists for HAR, and constructing one requires domain expertise without guarantee of consistency between experts due to a lack of objective rankings across classes e.g. is *running* more closely related to *walking* or *kicking*? To ensure a consistent and meaningful class structure, we propose deriving a hierarchy based on statistical features of the classes and self-supervised clustering, .

Organized in a hierarchy or not, feature extraction is a central factor for classification performance. In the broader field of AI, Large Language Models (LLMs) have become foundational for most tasks, including those outside of the target domain. Within HAR, attempts to apply pretrained LLMs for zero-shot prediction still underperform smaller dedicated models [38], and work on creating a dedicated LLM or transformer-based approach for HAR is still ongoing [59]. Convolutional networks remain state of the art in HAR, and like LLMs can be improved through transfer learning. We adapt the ResNet architecture from Yuan et al. [85], which is based on a single-device accelerometer, to a multi-device, multi-sensor set up and fine tune the resulting hierarchical architecture on our data.

To produce robust classification models, HAR datasets should satisfy two conditions : a high diversity of classes, and a high number of subjects. A high diversity of classes allows for in depth OOD analysis and comparison. A high number of subjects is required so that some subjects can be used as the test set, which is essential for generalising the model to actual applications. In this paper we introduce a new HAR dataset: NFI_FARED, containing data of multiple subjects performing nineteen activities from a broad spectrum of contexts, including daily living, commuting, and rapid movements.

Contributions of this study include:

1. An open-set activity classifier with both high in-distribution and out-of-distribution accuracy,

reflecting the practical requirements of HAR.[2]

2. A hierarchical model architecture which both improves performance, and provides insight to out-of-distribution samples.

3. NFI_FARED, a purpose-built HAR dataset containing body-worn sensor data for a diverse range of activities from many participants.

The paper is laid out as follows: in Section 1, we provide a brief overview of existing HAR techniques and OOD detection methods. Section 2 describes the proposed Hi-OSCAR method, including the model architecture training paradigm. Our new dataset NFI_FARED is presented in Section 3 and compared with a selection of already publicly available HAR datasets. Experimental setup and results are detailed in Section 4, with a discussion in Section 5. A brief summary, as well as potential future work and concluding remarks are provided in Section 6.

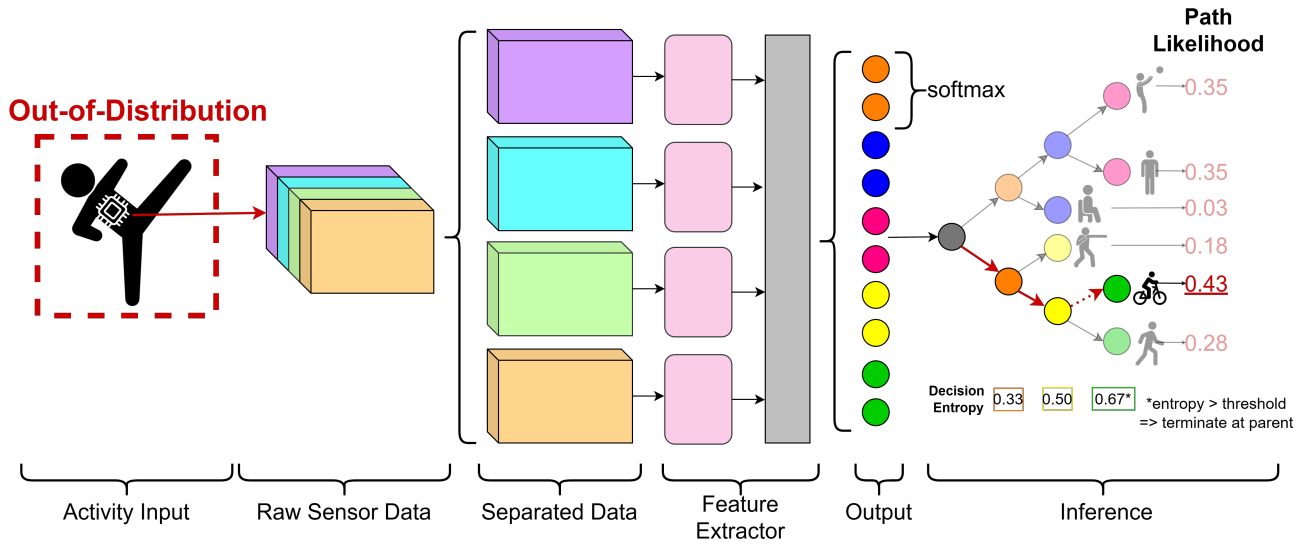


Figure 1: Model inference for an Out-of-Distribution activity using Hi-OSCAR. Raw sensor data are separated sensor-wise and fed through a feature extractor, giving an output vector of length $|H|$ (where $|H|$ ¿ number of classes). Elements of the output vector correspond with nodes in the hierarchy. The Path Likelihood of a class is computed by multiplying the softmax scores of each node along the path from root to node, and highest path likelihood determines the predicted path. If the decision entropy of any node along the chosen path exceeds the selected threshold, prediction is terminated. As an example here, the OOD class *kicking* would classified as *cycling* in the closed-set system, but due to high entropy it is classed as OOD. Prediction is also terminated at the penultimate node due to high decision entropy, indicating although it is OOD, it is similar to cycling and running.

1 Related Work

1.1 Human Activity Recognition

Early research in HAR relied on handcrafted feature extraction usually based upon either the time domain [14, 7, 43], the frequency domain [2, 87, 10], or both [6, 65, 64]. Handcrafted features are relatively straightforward and cheap, and can be made even more powerful when used with machine learning algorithms such as support vector machines [75], decision trees [58], k-nearest neighbours [54],

[2]Project GitHub: https://github.com/Con-or-McCarthy/Hi_OSCAR

random forests [8], or multilayer perceptrons [62]. Handcrafted features, however, are not necessarily effective, and approaches can be both expert- and task-specific and therefore do not generalize.

Deep Learning (DL) based methods for HAR have advanced the field massively in recent years. Convolutional Neural Networks (CNNs) process high dimensional data efficiently and can extract features directly from the time series. Combining CNNs with Recurrent Neural Networks (RNNs) to leverage the temporal aspects of the data has been an effective approach for classification. Examples are CNN layers combined with LSTMs, BiLSTMs, GRUs, and BiGRUs [60, 51] which are effective for classification, as well as other variations incorporating residual modules [32] and unique combinations of CNNs based on modality before combination and being fed into an LSTM [18]. To focus upon more important time points within the input sequence, Singh et al. [67] include a self-attention mechanism in a CNN-LSTM. Attention is also used with multilevel residual networks [1] and deep triplet networks [71]. However, when dealing with short windows of activities an RNN component is of limited use, since there is little time dependency within a five second span of a person driving a car, for example.

Given a well chosen backbone, an RNN component is not strictly necessary for SOTA results, CNN/ResNet alone can be used for classification, demonstrated by Mekruksavanich et al. [52], applying a ResNet-based model directly to activity time series, while Xu et al. [82] convert a HAR task to a typical computer vision problem by using the Gramian Angular Field algorithm to convert the 1D time series into 2D images and applied deep ResNets to the resulting images. Domain knowledge can also be leveraged, such as by using symbolic reasoning and contextual information alongside deep neural networks for prediction [5]. Though effective, these methods are often constrained to specific data forms and collection procedures.

1.2 Out-of-Distribution Detection

Open-set classifiers are typically set up as closed-set classifiers for classifying the ID samples with an added OOD detection component for distinguishing between ID and OOD samples which can then overrule the classification by flagging the sample as OOD. Post-hoc techniques are often the simplest to implement, since they require no adjustments to the underlying model, and can identify OOD samples by using a scoring function such as thresholding maximum softmax probability [33], energy score [49], GradNorm score [35], or fitting a distribution to the activations [13]. As straightforward scoring functions can have different effects depending on scenario and model setup they can complicate implementation. To improve OOD classification, the model activations themselves can also be adjusted at inference using ASH [24]. This involves simplifying the model representation during inference, compromising ID classification. Temperature scaling can also be used for calibration, for example ODIN [47], although this also requires perturbing inputs to further separate ID samples from OOD. Boyer et al. [15] evaluate many of these post hoc methods for HAR applications, and find that state-of-the-art OOD detection techniques are not effective in the HAR domain.

Instead of creating scoring functions based on model activations, training an OOD classifier on OOD samples can be a more direct approach. However, in the absence of OOD data during training (by definition), OOD samples must be simulated using the available training data. Outliers can be used as surrogate OOD examples, with output regularised to produce lower confidence on either manually collected outliers [34, 81], or on virtual outliers produced by the model [72, 26]. Both sources of outliers have drawbacks, either from the extra labour required, or from relying on the quality of the feature space. Nie et al. [57] circumvent this by directly perturbing input images in the pixel space, but this approach lacks a direct analogue in the HAR domain. Generative models can also be incorporated into the open-set classification framework using the assumption that generative models trained on OOD data will have more trouble reconstructing OOD samples than ID samples [30, 86, 61]. The associated reconstruction loss is used as an OOD scoring function. Tonmoy et al. [74] and Li et al. [46] both use this approach for open-set classification on HAR datasets. However, the assumption that OOD samples

are harder to reconstruct than ID may not always hold [56], and additionally, generative models are expensive and require increased testing time in the inference stage.

1.3 Public HAR Datasets

Due to the innumerable possible applications and useful scenarios for HAR, a great many datasets have been collected using assorted sensors, sizes, set ups, and purposes. A broad distinction can be drawn between vision-based and sensor-based datasets. Vision-based datasets [41, 44, 66] use input of videos of subjects performing physical activities, while sensor-based datasets provide input from body worn Inertial Measurement Unit (IMU) devices, usually containing accelerometer and gyroscope sensors, and also commonly containing some selection of temperature, heart rate, magnetometer and altitude sensors, among others.

IMU devices are cheap and non-invasive, making large-scale data collection relatively feasible, and projects such as the UK-Biobank [25] and NHANES [12] have produced huge amounts of sensor data. However, this data is unlabelled, which is the primary bottleneck in HAR data collection. As such, annotated datasets are generally small and tailored to a specific purpose. UniMiB-SHAR [53] and MobiAct [16] are two popular HAR datasets, collected for training fall detection models, passively identifying when a patient might have injured themselves. Their activity classes are split between activities of daily living (ADL) like sitting or walking, and different types of falls. Other HAR datasets such as MotionSense [50] were collected to investigate whether physical and demographic attributes can be inferred from sensor data, and REAL-DISP [11] looks at the impact of inconsistent sensor placement on the body. These datasets contain high quality HAR data, however the original focus of the collection process often means a significant portion must be discarded when utilising them for HAR classification. Information also must be discarded when multiple heterogenous devices are used, for example when using the dataset HHAR [68]. During the acquisition of this dataset, each subject wore eight smart phones and four smart watches, but due to differences in sampling rates and sensors between devices, for most use cases only a single device can be used for training/testing. Smart devices are still useful and popular for data collection thanks to their wide availability and application relevance. Two of the most commonly studied HAR datasets, WISDM [45] and UCI HAR [3], have annotated IMU data from smart phones for six classes of ADL, although they also suffer from inconsistent sampling across devices. The six classes of ADL: standing, sitting, laying down, walking, downstairs and upstairs are the same classes in almost all the above described datasets, with datasets rarely having any more than two classes not from this set of activities. This lack of activity diversity makes OOD examination by omitting classes during training difficult, since a single class represents a large portion of total data available. Some HAR datasets such as OPPORTUNITY do contain a NULL class alongside the defined activities, typically denoting unlabelled or unguided periods of activity. Since these periods are often inherently untracked and NULL is used to label all activities taking place during unscripted moments, it can occur that the NULL data is not truly distinct from the listed classes, which can lead to deflated or misleading results. Datasets containing typical ADL often have problematic overlap between ID and NULL classes [15, 22]. Therefore, while NULL classes can function as OOD sets, using omitted ID classes is preferable to ensure truly distinct sets of OOD activities.

For our analysis, we identified three datasets from the HAR community. Firstly, PAMAP [63] which is based on three IMUs on the wrist, chest, and ankle. The dataset has nine subjects, however only eight had complete data for all twelve activity classes. OPPORTUNITY [17] has four ADL from four subjects, each wearing seven IMUs on the arms, back, and legs. Participants wore twelve additional Bluetooth accelerometers, however they were not usable due to high levels of noise. Activities were annotated at different levels of specificity. Due to missing data on some subjects for higher levels of specificity, we target the locomotive level of activities, consisting of four classes of motion. Finally, RealWorld [70] is the largest of the three datasets in terms of length (18 hours). Fifteen subjects wore seven mobile devices (six smartphones and a smartwatch) located on the head, torso, arms, and legs.

Subjects performed eight different activities for roughly ten minutes per activity, with the exception of jumping. For all three datasets the IMUs contained accelerometer, gyroscope, and magnetometer sensors.

2 Method

We will first give an overview of our approach. Hi-OSCAR takes as input raw sensor signals collected from one or more IMU devices which are fed into a ResNet-based feature extractor (Section 2.1). The output vector is of size $|H|$, the number of nodes in the hierarchy, determined by the generation process, where $|H| > |K|$, the number of ID classes in the dataset. The hierarchy H is generated using self-supervised Hierarchical Agglomerative Clustering (HAC). Leaf nodes correspond directly to classes in the dataset, and the internal nodes correspond to implicit activity categories that emerge from the hierarchical relationships between the labelled classes, where their semantic interpretation is derived from the known activities they encompass (Section 2.2). Each class $k \in K \subseteq H$ has an associated path likelihood used to classify the sample (Section 2.4). If along the prediction path, decisions between nodes have high enough decision entropy ($\equiv$ low prediction confidence) on average, they are labelled OOD (Section 2.5). During inference on OOD samples, sufficiently high entropy on a node during traversal will trigger the inference stopping criterion, terminating prediction at this step and returning the parent node as the final prediction (Section 2.6). This allows the model to still provide meaningful information to the user when the class cannot be determined (Section 2.7). During training, it is therefore also necessary to temper overconfidence in predictions by simulating OOD data using only ID classes from the training set (Section 2.3). An overview of the procedure is given in Figure 1.

2.1 Feature Extractor

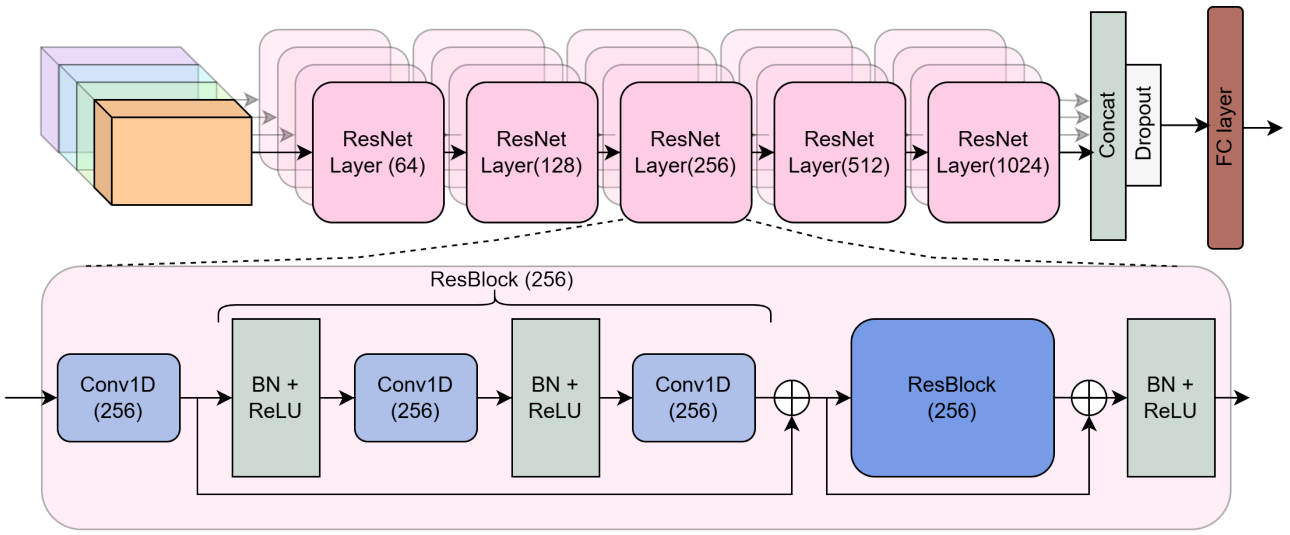


Figure 2: Feature extractor architecture adapted from Yuan et al. [85], extended to the multi-sensor, multi-device domain.

The feature extractor component is purely CNN-based, as in Mekruksavanich et al. [52], and is adapted from Yuan et al. [85]. Raw time series input is separated at the sensor level and fed through the feature extractor in parallel. As shown in Figure 2, each parallel component consists of five successive ResNet layers, increasing the number of channels up to 1024. The original architecture was designed for a single accelerometer device. The feature extractor has been duplicated along the sensor axis to

permit parallel inputs from multiple devices and sensors, with input channels also adjusted to accept non-triaxial sensor input. Each parallel feature extractor is also initialised with the pretrained weights from [85]. The output of the final ResNet layer is concatenated to the outputs of the other parallel blocks followed by a dropout operation. A Fully Connected (FC) layer then outputs a vector of length $|H|$ containing probabilities for every node in H. During training, all the weights of the ResNet layers are fully trainable and fine tuned, while the fully connected layers are simultaneously trained from scratch.

2.2 Hierarchy Generation

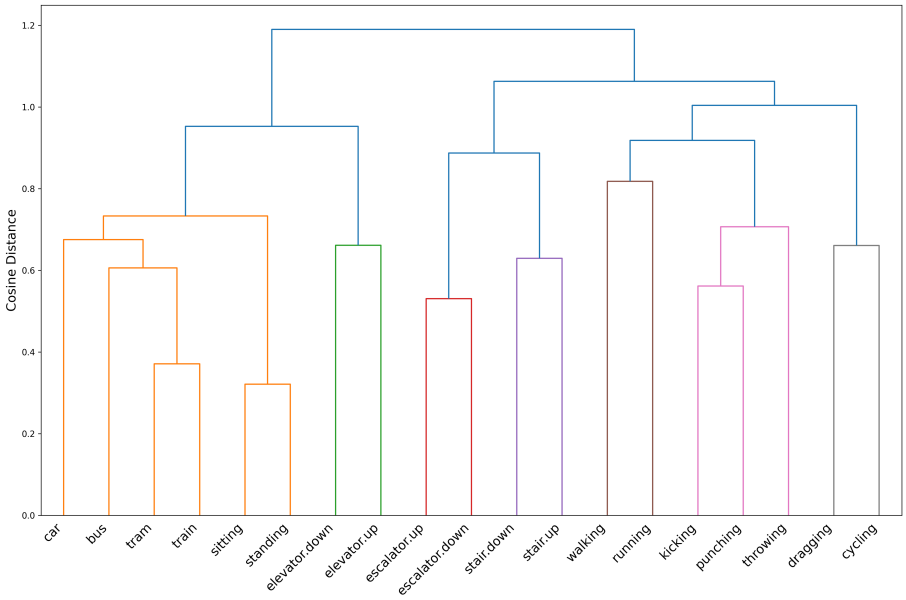
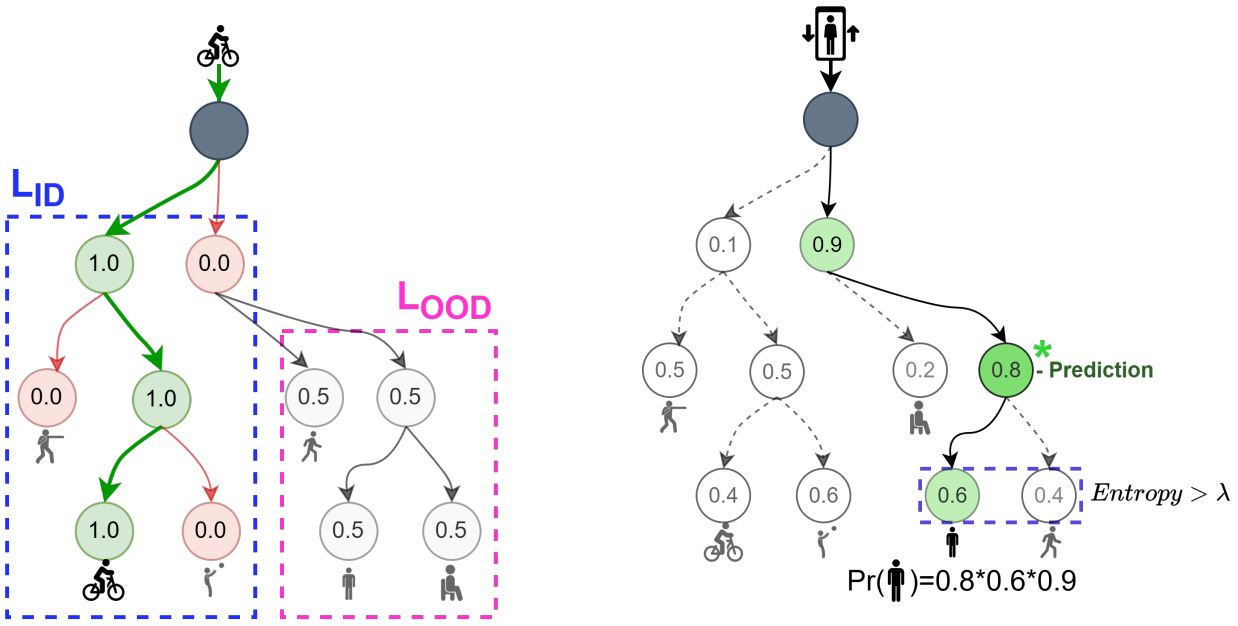


Figure 3: Dendrogram hierarchy generated using Hierarchical Agglomerative Clustering on NFI_FARED with total Cosine Distance after each merge indicated on the y-axis.

Class hierarchies outside of HAR are typically constructed from domain knowledge and ontologies. No definitive hierarchy exists for HAR classes as constructing one is difficult due to a lack of objective rankings of activity classes meaning expert opinions can differ significantly. One problem specific to the HAR domain is the mismatch between semantically related activities and physically related activities. Two activities, such as *standing* and *elevator* can be completely distinct semantically despite being almost identical in terms of physical performance. When incorporating a hierarchy into a classification system, it is preferable to emphasise physical similarities which are present in the data over semantic ones, which are not. As such, deriving a hierarchy with unsupervised clustering such as Hierarchical Agglomerative Clustering (HAC) is better suited to HAR than using pre-defined ontologies. HAC takes a bottom up approach, each class starts in its own cluster and pairs of clusters are merged until all classes have been combined into the Root.

To create the hierarchy, firstly the feature extractor from Section 2.1 is trained for five epochs on the training set. The centroids of the classes in the embedding space are then used to form the initial K clusters. The two closest clusters by cosine distance are merged. After merging the two closest clusters, the process is repeated until all classes are within a single cluster. The final output is a complete hierarchy of $|H|$ nodes, including all K classes as leaf nodes. An example of such a hierarchy generated from NFI_FARED using all available classes is shown in Figure 3, including the total cosine distance present after each merge.

2.3 Loss Function



(a) Training (b) Inference

Figure 4: Illustrative example of training and inference on the Hi-OSCAR hierarchy. (a) Visualisation of optimal activations during training. Nodes along the path to correct ID class are optimised by L_{ID}, while nodes off the path are optimised to the uniform distribution by L_{OOD} (b) Example visualisation of inference on OOD class *elevator*, showing example node activations. Standing has the highest path probability, however due to high entropy in the final level, an internal node is outputted.

Given Hierarchy $H = \{0, 1, ..., |H|-1\}$ generated by HAC, there are $|K|$ leaf nodes, corresponding to classes in the training set, and $|H|-|K|$ internal nodes. For each class node $k \in K$, there exists a unique path $anc(k)$ from the Root node to k (green path in Figure 4a). The loss function should encourage each node $n \in anc(k)$ to have an activation of 1, and its sibling to have activation 0, visualised in the blue box in Figure 4a (L_{ID}). When nodes are neither in $anc(k)$ or a sibling of a node in $anc(k)$, the loss function should maximise entropy at this split. Given that HAC produces trees with all binary splits, this is achieved when the sibling nodes each have activation of 0.5, shown in the pink box in Figure 4a (L_{OOD}). The computation of the two loss functions, L_{ID} and L_{OOD}, are described below.

Given true class $y \in K$, L_{ID} is the cumulative Cross Entropy (CE) loss of each node in $anc(y)$:

$$L_{ID} = \mathbf{W}_y \cdot \sum_{n \in \{anc(y), y\}} CE[p(n), \mathbf{y}] \tag{1}$$

where $\mathbf{W}_y$ is the inverse frequency of class y. This weighting term helps account for class imbalances within the training set.

The role of L_{OOD} is to maximise entropy (minimise confidence) of any split in the hierarchy not containing a node in $anc(y)$. This uses ID samples as pseudo-OOD for all nodes not intersecting with $anc(y)$. These nodes $n \notin anc(y)$ are optimised to the Uniform distribution $\mathbf{U}$ using Kullback-Leibler divergence:

$$L_{OOD} = \sum_{n \notin anc(y)} KL[p(n), \mathbf{U}]. \tag{2}$$

The two losses are then combined to produce the final loss function:

$$L = L_{ID} + L_{OOD} \tag{3}$$

2.4 Hierarchical Classification

With Hierarchy $H = \{0, 1, ..., |H| - 1\}$ generated by HAC, each of the $|H|$ output nodes of the model are assigned to a node, either internal or class. Softmax is applied to each pair of sibling nodes, so that for any given parent node, the activations of its children are normalised and sum to one. For each class $k \in K$, there exists a unique path from root node to class node $anc(k)$. This path likelihood is calculated by multiplying the softmax score of each node along the path $anc(k)$:

$$Pr(k) = p(k) \cdot \prod_{n \in anc(k)} p(n) \tag{4}$$

where $p(n)$ is the softmax score of node n. During inference, the path likelihood of all classes is calculated and the provisional predicted class is assigned the class with maximum value: $\hat{k} = max_{k \in K} Pr(k)$. Calculating total path likelihood for every class k rather than greedily choosing the maximum score at each node allows for mistakes earlier in the hierarchy to be corrected downstream [78]. Example path likelihood calculation is shown in Figure 4b.

2.5 OOD score

Once the provisional predicted class $\hat{k}$ has been chosen, the path $anc(\hat{k})$ is then traversed to determine if the sample is OOD. For this, we employ the entropy of each decision along the path $anc(\hat{k})$. At each node $n \in anc(\hat{k})$, the entropy of its children is calculated:

$$\mathbf{H}(n) = \sum_{n' \in ch(n)} -p(n')\log(p(n')). \tag{5}$$

We utilise mean path entropy $\mathbf{H}_{mean}$ as the OOD score, calculated as:

$$\mathbf{H}_{mean}(\hat{k}) = \frac{1}{len(anc(\hat{k}))} \sum_{n \in anc(\hat{k})} \mathbf{H}(n). \tag{6}$$

The higher the value of $\mathbf{H}_{mean}(\hat{k})$, the less confidence there is in the prediction, and the more likely it is to be OOD.

2.6 Inference Stopping Criterion

Beyond a simple accept/reject decision for OOD samples, Hi-OSCAR returns informative labels in the form of internal node predictions, providing richer OOD classifications which embed information about the most closely related classes. For every ID sample, the entropy $\mathbf{H}(n)$ of each node pairing $n \in anc(\hat{k})$ is calculated and stored. A threshold for stopping inference (λ) is chosen based on the values within the training set, e.g. choosing the 99^{th} percentile would result in inference stopping whenever a node's entropy is higher than 99% of the entropies recorded by that node in the ID data. We denote this by $\hat{\lambda} = 0.99$. Figure 4b illustrates an example of the inference stopping criterion being exceeded and outputting an internal node for an OOD sample.

The choice of $\hat{\lambda}$ depends on the chosen application's fidelity requirements. Higher values of $\hat{\lambda}$ will be more prone to 'over-prediction', i.e. terminating at a leaf node or deep in the tree, while lower values of $\hat{\lambda}$ will give coarser, less precise predictions of nodes closer to the Root.

The returned node for an OOD sample should be close to the leaf nodes of similar activities in the training set, informing the user about its dynamics, and the closer the predicted internal node is to a leaf nodes (measured by cosine distance), the more similar it is. Conversely, if the OOD sample is completely novel relative to the training set, the internal node predicted will be further from the leaf nodes, and closer to the Root of the hierarchy.

2.7 Prediction Evaluation

Interpretation of the internal node OOD predictions requires a measure of distance to determine how 'close' or 'far' the prediction is from the ID classes. The generated hierarchy H can be used for this purpose, since it already has grouped the classes and internal nodes based on similarity. Hierarchical distance, counting the number of edges between two nodes, can be used in this case. However, it assigns equal value to all edges, ignoring the differences in cosine distance associated with each merge. This can lead to inflated hierarchical distances for some classes when many similar activities are present in the dataset. Consider in Figure 3, *train* and *car* are separated by four edges, the same as *throwing* and *walking*, despite the former pair being closer in terms of cosine distance on the Y-axis.

We therefore propose weighting each edge by the increase in total cosine distance added by that merge during the HAC process, terming it Cumulative Cosine Distance. Given predicted node s and target node t with $lca(s,t) = q$, Cumulative Cosine Distance is calculated as:

$$D(s,t) = \sum_{\substack{u \supset s \\ u \subset q}} d(u,s) + \sum_{\substack{v \supset t \\ v \subset q}} d(v,t), \tag{7}$$

where $d(u,v)$ is the cosine distance. Lower values indicate the nodes are close i.e. similar in nature.

3 Dataset NFI_FARED

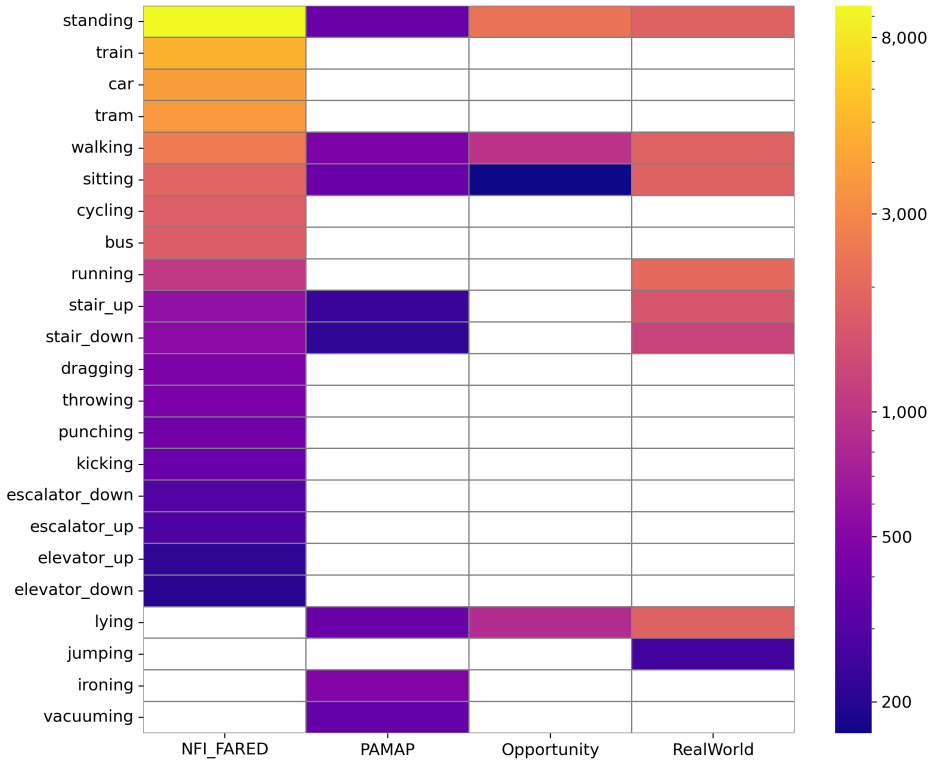


Figure 5: Number of 10 second samples from each dataset. Blank squares denote the activities not in that dataset.

The NFI_FARED dataset contains data from eleven participants wearing multiple devices and sensors. The dataset contains a diverse range of activities roughly grouped as everyday movements, elevation change, dynamic movements, and transportation, totalling 19 activities for each subject. The

Model	#Subjects	#IMUs	#Activities	#Hours	Sensors
PAMAP	8	3	8	11	accelerometer, gyroscope, magnetometer
Opportunity	4	5	4	7	accelerometer, gyroscope, magnetometer
RealWorld	15	5	8	18	accelerometer, gyroscope, magnetometer
NFI_FARED	11	2	19	40	accelerometer, gyroscope, altimeter

Table 1: Details of datasets used in analysis

dataset consists of a large amount of data, totalling 39 hours 48 minutes (14,267,742 samples). The selection of activities, as well as their abundance relative to other publicly available HAR datasets, is shown in Figure 5.

Below we detail the procedure followed to collect the NFI_FARED, as well as details on pre-processing.

3.1 Participants

Fourteen Participants are included (age 26.6 ± 8.8y), made up of 6 females and 8 males. Each subject signed an informed consent form, agreeing to the collection and spreading of their data anonymously. Height and weight data of each subject cannot be shared for privacy reasons. For each subject, data was collected during multiple sessions spread over several days. Due to scheduling conflicts, three participants could not attend all sessions, and three different subjects completed those sessions in their place. Their data was combined, providing effectively eleven participants worth of data.

3.2 Sensors

The two deployed IMUs measure linear acceleration with triaxial accelerometer (freq = 100 Hz), angular velocities with a triaxial gyroscope (freq = 100 Hz), air pressure with an altimeter (freq = 1 Hz), and temperature (freq = 1 Hz). The inclusion of air pressure data, a rarely utilised sensor in this domain, is notable. Air pressure is naturally informative for the many activities which include elevation changes. It is also a one dimensional signal, unlike acceleration and rotation, which occur in three dimensions. During analysis, the temperature signal was omitted since it is not related to body kinematics. This resulted in thirteen channels of data used in the analysis.

The IMU devices were placed on the lower back and on the dominant forearm of the subjects. The participants attached the devices themselves to introduce some heterogeneity in sensor placement.

3.3 Protocol

The dataset contains 19 activities, listed in Figure 5. The activities were recorded in four separate measurement sessions. Session one includes everyday activities, session two recorded elevation change activities, session three recorded dynamic activities, and session four recorded transport activities. The sequence of performed activities within each session was randomised for each subject.

Sessions one, two, and three were performed under controlled conditions. In between each activity the subject would rest one minute by either sitting or standing, as instructed by the instructor, who recorded the activities. It was found that sitting on a chair with a backrest influenced the accelerations recorded by the IMU on the back, so sitting breaks of both kinds (with and without backrest) were recorded for each subject. To encourage diversity in their execution, while performing the activities, the subjects were given as few instructions as possible. At the end of each session, the subject completed a "free-living" section, in which they would perform the activities from that session in any order and for

however long they wished without breaks. For both of these scenarios, the start and end times of each activity would be recorded. Here we describe the data collection process for each session:

Everyday Movements Session The movements walking, running and cycling were performed multiple times for known distances. For running and walking there are two distances: ±240m and ±90m, each performed twice. For cycling, the subjects cycled a set number of loop around the car park of the experimental location three times. The number of loops each were varied between two, three, and four. During the static sitting movement, the participant sat on a bench without backrest.

Elevation Change Session The movements stair, escalator and elevator were performed four times each going upstairs and downstairs. The stair climbing and escalator movements were performed at different velocities. For stair climbing the subject was instructed to walk or to run the stairs. For the escalator, the subject was instructed to stand or walk on the escalator. After each dynamic movement the participant would either stand or sit for one minute at the end of the stairs/escalator. During the static sitting movement, they sat on a bench with backrest.

Dynamic Movements Session The dynamic movements: kicking, throwing, punching and dragging were done four times for different time intervals: 10 seconds, 20 seconds and two times 30 seconds. During the kicking and punching movement, the participant would kick or punch, respectively, a punching bag. The kicking/punching was alternated between their dominant and non-dominant leg/arm. For the throwing task, the subject threw a ball using their dominant arm down a hallway to the another person, who rolled it back to the subject to pick it up and throw again until the time was complete. The ball weighed 1 kg. For the dragging activity the subject dragged the punching bag weighing 34 kg through a hallway.

Transportation Session In session four, the subjects wore both IMUs while travelling home. The transport modes of riding the train, tram, and in a car were being recorded. If the subject had not used one or more of the listed modes of transport in their trip home, arrangements were made to ensure data for all transport types were recorded for each individual e.g. if they had not travelled by bus or car, they were then driven to the bus station, from which they would ride a bus back to an agreed location to return the IMUs. The participants recorded the start and end times of each travel activity by themselves.

3.4 Data acquisition

The starting times of the measurement for both devices were synchronized. The raw signal data from the IMUs were pre-processed using proprietary scripts. The data was labelled by matching the timestamps of activities recorded by the instructor with the timestamps of the raw signals. During the transport session, the data was self-reported by the subject making a note of the start and end time of each transport type. Where there was misalignment in the labels as a result of the self-labelling, e.g. a subject's watch being set wrong, labels were adjusted by a minute or two, using the signals of the sensors to find the actual transition moment between activities. It was found to be possible to locate these transition moments by eye from the sensor data, with clear indications of activity changes that aligned with the duration of activity reported by the subject, only with the start and end times shifted. Given the amount of transport data collected, this adjustment only affected a small minority of the total data. Accelerometer, gyroscope, and air pressure data are included in the dataset. The air pressure data was linearly interpolated to the frequency of 100Hz in order to match the frequency of the other signals. The data is fully publicly available for download, and scripts for processing data into windows are provided in the project GitHub.

4 Experiments

To assess open-set classification performance we test on both ID and OOD tasks (Sections 4.1 and 4.2, respectively), comparing against state-of-the-art baselines from HAR specific methods and generic Machine Learning methods. In Section 4.3, we investigate the differences in performance for different OOD classes, and in Section 4.4 the impact of threshold $\hat{\lambda}$ on prediction quality. Section 4.5 analyses the effect of different hierarchy generation methods, both data- and knowledge-driven, and Section 4.7 looks at the influence of window size upon performance.

All experiments were performed using k-fold cross-validation on the subject level. Datasets NFI_FARED and RealWorld had two subjects in each fold, while OPPORTUNITY and PAMAP had one, due to the number of subjects available in the datasets. Data was split subject-wise to prevent data leakage and to better evaluate generalisability to new subjects' data. The hierarchy used by Hi-OSCAR was always generated using only samples from the training set to further prevent data leakage. Each fold was then repeated five times, reporting the mean and standard deviation (in brackets). The data was divided into 10s windows with 50% overlap and downsampled to 30 Hz. Training was run for a maximum of 250 epochs using early stopping, which terminated training if validation loss did not decrease for 5 epochs in a row, to prevent overfitting.

Hi-OSCAR was trained with the Adam optimiser with learning rate 1e-4, betas = (0.9,0.999). All experiments were performed using an NVIDIA RTX 4500 Ada Generation GPU. Baseline models used for comparison were replicated from the original authors' descriptions and code (if available) and implemented in PyTorch. For baselines not taken from literature or when hyperparameters were not specified, hyperparameter tuning was used to set values. Further details are provided in Appendix A.1. To enable fair and meaningful comparisons between models, we standardised the evaluation protocol (cross-validation, window size, dataset choice, etc.). As a consequence of standardisation, there are some differences in performance between the reported results in the original papers and those in our experiments. For example, the popular LIMU-GRU model [83] uses 95% of the training data for unsupervised learning to generate embeddings, and the remaining 5% for supervised training. In our set up, all baselines utilise 100% of the data, making fair comparison with LIMU-GRU not possible. Instead we include the GRU baseline, which is comparable to the classifier used in LIMU-GRU when pretraining is not performed beforehand.

4.1 Activity Classification

We evaluate ID classification using NFI_FARED and the three public HAR datasets: PAMAP, OPPORTUNITY, and RealWorld. This first experiment looks at pure ID performance when all classes are present during training.

Performance is measured using the following metric:

- **Macro F1-score** is calculated by finding the F1-score $F1 = \frac{2 \times TP_{ID}}{2 \times TP_{ID} + FP_{ID} + FN_{ID}}$ for each class and computing the unweighted mean. Macro-F1 score assigns equal importance to all classes. An ID True Positive (TP_{ID}) is an ID sample correctly labelled and other quantities are similarly defined.

Results of the ID experiments are shown in Table 2. Hi-OSCAR performs best on all four datasets, showing a 1.76%, 0.85%, and 1.57% improvement compared to baselines on datasets PAMAP, OPPORTUNITY, and RealWorld, respectively. Significance was assessed using paired t-tests over the folds. They show that although worse on average, Random Forest did not perform significantly worse than Hi-OSCAR on the three public HAR datasets. CNNBiGRU and CNNLSTM-Att also showed no significant difference to Hi-OSCAR on PAMAP and OPPORTUNITY, however this can also be attributed to the high variance in their outputs, supported by a large effect size (Cohen's distance) relative to Hi-OSCAR of -1.32 and -1.2 on PAMAP for CNNLSTM-Att and CNNBiGRU, respectively, compared to -0.8 for

	PAMAP	OPPORTUNITY	RealWorld	NFI_FARED
Model	F1 ↑	F1 ↑	F1 ↑	F1 ↑
GRU	0.366 (0.081)	0.422 (0.051)	0.384 (0.036)	0.169 (0.025)
DCNN [21]	0.609 (0.083)	0.257 (0.102)	0.226 (0.165)	0.223 (0.088)
Transformer [23]	0.456 (0.131)	0.579 (0.057)	0.597 (0.111)	0.235 (0.056)
DeepSense [84]	0.657 (0.069)	0.551 (0.110)	0.663 (0.073)	0.334 (0.024)
ResNet-SE [52]	0.534 (0.077)	0.527 (0.067)	0.647 (0.092)	0.442 (0.069)
CNNLSTM-Att [55]	0.428 (0.342)	0.580 (0.113)	0.586 (0.150)	0.484 (0.101)
MLP	0.759 (0.094)	0.727 (0.031)	0.760 (0.059)	0.502 (0.048)
CNNBiGRU [51]	0.823 (0.140)	0.761 (0.121)	0.772 (0.071)	0.584 (0.155)
Random Forest	0.911 (0.036)	0.820 (0.038)	0.889 (0.051)	0.677 (0.043)
Hi-OSCAR	**0.927** (0.055)	**0.827** (0.027)	**0.903** (0.028)	**0.827** (0.047)

Table 2: F1 scores for ID classification on HAR datasets. Standard deviation is shown in brackets and best performing model on each dataset is in bold. Underlined scores indicate models whose performance is not statistically significantly different ($p \geq 0.05$) from the best-performing model, according to a paired t-test over the folds.

Random Forest. Similarly, on OPPORTUNITY, CNNBiGRU has an effect size of -0.7 compared to Random Forest's -0.2.

On NFI_FARED, Hi-OSCAR beats baselines by over 20%. All baselines struggled with the more complex task of classifying 19 activities. Random Forest, the best baseline, saw a 17% drop in score between NFI_FARED and OPPORTUNITY, its second worst dataset. Meanwhile, Hi-OSCAR scored the same as it did on the OPPORTUNITY dataset.

4.2 Out-of-Distribution Detection

Open-set classification was evaluated using the datasets NFI_FARED and OPPORTUNITY. For NFI_FARED, one class (the OOD class) was omitted from the training set before training under the same conditions as in Section 4.1. For OPPORTUNITY, the provided NULL class was used as OOD, and also not included in the training set. In both cases, the test set included all classes, including the OOD class. An OOD score was calculated for every sample in the test set, both ID and OOD. This OOD score is used to distinguish samples of ID classes from samples from the OOD class, where a larger OOD score means a sample is more likely OOD. Choice of OOD score depends on method. In the context of OOD detection, a True Positive (TP_{OOD}) is defined as a correctly labelled OOD sample, a False Positive (FP_{OOD}) is an ID sample incorrectly labelled OOD, and so on for the other quantities. How effective the OOD score was at correctly labelling OOD samples was measured using the following metrics:

- **AUROC** Is the Area Under the Receiver Operating Curve. The Receiver Operating Characteristic (ROC) curve plots the relationship between TPR and FPR. The area under the ROC curve can be interpreted as the probability that a positive example (ID) will have a higher detection score than a negative example (OOD). Consequently, a random positive example detector corresponds to a 0.5 AUROC, and a "perfect" classifier corresponds to 1.
- **Detection Error** measures the misclassification probability when TPR is 95%. The definition of Detection Error is given by Detection Error $= 0.5(FPR_{OOD} + FNR_{OOD})$.

We compare Hi-OSCAR to a number of baseline open-set classification methods, namely OpenNet [31], k-Nearest-Neighbours (kNN), and Variational Auto Encoder (VAE). All methods use the same feature extractor specified in Section 2.1 and otherwise use the training procedure required by the

			OOD		ID
Model	Dataset	OOD class	AUROC ↑	Detection Error ↓	F1 ↑
OpenNet [31]	NFI_FARED	dragging	0.631 (0.122)	0.343 (0.089)	0.734 (0.019)
		tram	**0.695** (0.097)	**0.339** (0.070)	0.759 (0.057)
		running	**0.731** (0.184)	**0.234** (0.123)	0.685 (0.061)
	OPPORTUNITY	NULL	0.538 (0.179)	0.337 (0.120)	0.796 (0.023)
VAE	NFI_FARED	dragging	0.471 (0.119)	0.405 (0.034)	0.593 (0.042)
		tram	0.506 (0.144)	0.410 (0.068)	0.599 (0.046)
		running	0.545 (0.164)	0.317 (0.072)	0.604 (0.047)
	OPPORTUNITY	NULL	0.451 (0.150)	0.380 (0.051)	0.505 (0.087)
kNN	NFI_FARED	dragging	0.693 (0.131)	0.246 (0.022)	0.813 (0.060)
		tram	0.465 (0.068)	0.445 (0.044)	**0.844** (0.049)
		running	0.614 (0.127)	0.293 (0.087)	**0.816** (0.064)
	OPPORTUNITY	NULL	0.456 (0.177)	0.406 (0.082)	0.838 (0.037)
Hi-OSCAR	NFI_FARED	dragging	**0.898** (0.067)	**0.159** (0.076)	**0.815** (0.051)
		tram	0.659 (0.096)	0.353 (0.054)	0.836 (0.023)
		running	0.295 (0.144)	0.460 (0.028)	**0.824** (0.050)
	OPPORTUNITY	NULL	**0.820** (0.048)	**0.188** (0.042)	0.828 (0.050)

Table 3: Comparison with open-set baselines, using OOD metrics AUROC and Detection Error and ID metric F1. Models are trained and tested using classes *dragging*, *running*, and *tram* from NFI_FARED. F1 is calculated based on remaining ID classes. Best results are in bold. Underlined scores indicate models whose performance is not statistically significantly different ($p \geq 0.05$) from the best-performing model, according to a paired t-test over the folds.

method, e.g. a different loss function for OpenNet. Varying only the OOD detection method is done to evaluate the contribution of the hierarchical structure to OOD performance. Such changes to the model or training routine can also impact ID performance. We therefore also include the ID metric macro-F1 score in the results (Table 3). Though the focus of these experiments is on OOD performance, it is important to see how it balances with ID results for reliable open-set classification.

Experiments are run using three different choices of OOD class, namely *dragging*, *tram*, and *running*. These classes cover the range of Hi-OSCAR's performance and give a fuller picture of OOD ability (see Section 4.3).

When *dragging* is the OOD class, Hi-OSCAR performs best, beating the best baselines kNN by 30% on AUROC 35% on detection error. For *tram*, OpenNet is best, scoring 5% higher than Hi-OSCAR on AUROC and 4% higher on detection error. According to paired t-tests, OpenNet and Hi-OSCAR were statistically equivalent for OOD class *tram*. On *running*, Hi-OSCAR scores worse than the baselines with AUROC 0.295, far below the best baseline OpenNet. For the NULL class from OPPORTUNITY, Hi-OSCAR scores higher than all the baselines on both AUROC and detection error, beating next best model OpenNet by 52% and 44%, respectively.

ID performance varied across the baselines, and is reduced for both OpenNet and VAE, whose training procedures negatively impact performance. kNN, the least invasive method, is statistically equivalent to Hi-OSCAR on F1 score. While OpenNet and kNN match Hi-OSCAR's performance on ID and OOD tasks separately, only Hi-OSCAR is competitive on both dimensions.

For all methods, OOD performance varies significantly with choice of OOD class. Each of the three examined OOD classes proves most difficult (measured by AUROC) for at least one of the methods. Similarly, the easiest to flag OOD class varies depending on method. Experiments using OOD sets containing multiple randomly selected classes in the OOD sets are included in Appendix A.3.

4.3 Effect of Activity on Out-of-Distribution Performance

	OOD		ID
Excluded Class	AUROC ↑	Detection Error ↓	F1 ↑
dragging	0.898 (0.067)	0.159 (0.076)	0.815 (0.051)
cycling	0.887 (0.063)	0.184 (0.068)	0.820 (0.036)
standing	0.865 (0.051)	0.207 (0.052)	0.834 (0.038)
throwing	0.783 (0.085)	0.250 (0.059)	0.812 (0.049)
sitting	0.777 (0.088)	0.253 (0.100)	0.844 (0.038)
train	0.761 (0.086)	0.286 (0.046)	0.845 (0.032)
escalator.down	0.736 (0.088)	0.297 (0.057)	0.838 (0.045)
bus	0.735 (0.099)	0.308 (0.067)	0.835 (0.060)
tram	0.659 (0.096)	0.353 (0.054)	0.836 (0.023)
stair.up	0.641 (0.129)	0.364 (0.086)	0.825 (0.050)
elevator.down	0.617 (0.059)	0.358 (0.046)	0.834 (0.028)
car	0.597 (0.144)	0.389 (0.061)	0.827 (0.049)
kicking	0.591 (0.304)	0.250 (0.132)	0.840 (0.032)
elevator.up	0.572 (0.094)	0.376 (0.023)	0.838 (0.041)
escalator.up	0.552 (0.124)	0.396 (0.073)	0.839 (0.024)
stair.down	0.546 (0.238)	0.383 (0.114)	0.813 (0.039)
walking	0.524 (0.119)	0.384 (0.072)	0.853 (0.026)
punching	0.503 (0.171)	0.406 (0.085)	0.818 (0.027)
running	0.295 (0.144)	0.460 (0.028)	0.824 (0.050)

Table 4: Open-Set classification results for different selections of OOD class.

To investigate the impact of OOD class on open-set performance, we show the results for Hi-OSCAR on each class of NFI_FARED in Table 4. NFI_FARED contains a high number of classes compared with existing HAR datasets, and notably, a number of those classes are similar to each other. Overlap or equivalence between classes increases the difficulty of OOD classification for these similar classes.

OOD results show significant variation. Best performing classes *dragging*, *cycling*, and *standing* all score above 0.85 on AUROC. Meanwhile, worst performing classes *running*, walking, *punching*, and *stair down* all score below 0.55. Notably, these low scoring classes all have similar activity classes in NFI_FARED. Across all OOD classes, ID performance is comparatively consistent, with macro-F1 score remaining between 0.812 (*throwing*) and 0.853 (*walking*).

We compare variations in OOD performance with the best performing OOD baseline, OpenNet in Figure 6. Hi-OSCAR performs better on 13 out of 20 activities (including NULL from OPPORTUNITY), and is superior on average. Distribution of OOD performance between the methods on many classes is roughly in line for a number of classes, but there are significant differences in some cases.

In order to investigate the cause of the gaps in OOD performance, we applied the eXplainable AI (XAI) technique Integrated Gradients [69] to both Hi-OSCAR and OpenNet. Outputs of Integrated Gradients are shown in Figure 7 for classes *cycling* and *running*, the two classes with the largest gap in AUROC in favour of either method. For all cases, the back acceleration sensor is important, and aside from OpenNet: *cycling*, is the most important sensor overall. For Hi-OSCAR, back acceleration and arm acceleration were the two most important sensors, with less contribution coming from back and arm rotation or atmospheric pressure. For OpenNet, back acceleration and back rotation had the largest contributions, with back rotation being the most important for OOD class *cycling*. Interestingly, this is the OOD class OpenNet performed worst on, relative to Hi-OSCAR.

Despite using the same feature extractor, the two methods focus on different parts of input to

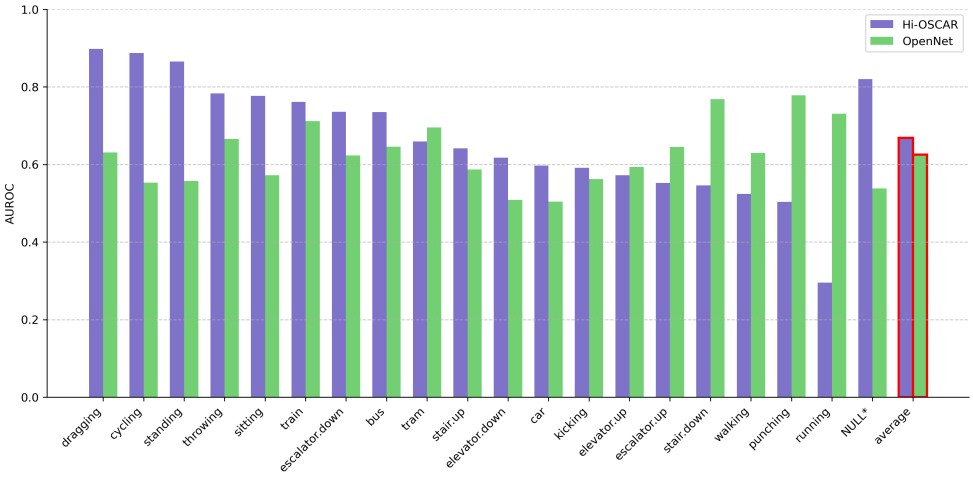


Figure 6: Comparison of AUROC between Hi-OSCAR and OpenNet for different OOD classes from dataset NFI_FARED. *NULL class is from the OPPORTUNITY dataset

determine output. Not only are different sections of the input data used more, but across the sensors themselves, Hi-OSCAR looks more at acceleration from both device placements, while OpenNet uses more acceleration and rotation from the back-placed device. These differences influence the variations in OOD performance seen in Figure 6.

4.4 Effect of Threshold on Approximating OOD Class

An added benefit of the hierarchical output is the ability to predict internal nodes for approximating OOD classes. In this experiment, a hierarchy was generated using all classes in NFI_FARED, so that the position of the predicted internal nodes could be compared to where the OOD class "should" be. The measure of the quality of a prediction is how close the predicted node is to the true position of the OOD class in the hierarchy. This closeness is measured using Cumulative Cosine Distance (Equation 7), where lower distances indicate the predicted node for the OOD is close to its true position.

Figure 8a plots threshold $\hat{\lambda}$ against average Cumulative Cosine Distance across all classes, as well as three highlighted classes *dragging*, *running*, and *tram*. High values of $\hat{\lambda}$ return predictions closer to leaf nodes, and as it is reduced, traversal is more likely to terminate earlier up the tree, closer to the root. The blue line in Figure 8 showing average Cumulative Cosine Distance shows a steady drop from $\hat{\lambda} = 1.00$ to $\hat{\lambda} = 0.85$ before increasing as $\hat{\lambda}$ is reduced further. This corresponds to "overprediction" when $\hat{\lambda}$ is at a maximum, that is, going past the OOD class towards another leaf node, then as $\hat{\lambda}$ is reduced somewhat, the prediction improves, moving up a level or two, closer to the true prediction. As $\hat{\lambda}$ decreases further, predictions continue getting further away from the true position. This is illustrated for OOD class *running* in Figure 8b.

All three plotted classes exhibit an initial decrease before steady increase in distance from true class. The class *tram* follows the shape of the average curve closely, albeit with a flatter portion between $\hat{\lambda} = 0.85$ and $\hat{\lambda} = 0.6$, indicating that predictions do not improve or disimprove much in this range. Class *running*, also shows an improvement in prediction quality as $\hat{\lambda}$ moves from 1.0 to 0.95, before a sharp increase. Notably, *running* was the worst classified OOD class in Section 4.3, and here the cosine distance to the true position is very low, revealing that running is being incorrectly classified as a very similar class. *Dragging* is actually closer to the true position when $\hat{\lambda} = 0$ than it is for $\hat{\lambda} = 1$. This is an example of when the 'true' class is closer to the Root than nearby leaf nodes, still the minimum distance to the true class occurs at $\hat{\lambda} = 0.85$. Although the curves are show some differences in shape and minimum, setting $\hat{\lambda} = 0.85$, the optimal setting on average, for all curves returns predictions close

Hierarchy Type	AUROC ↑	Detection Error ↓	F1 ↑
Chat-GPT 4.5*	0.624 (0.099)	0.363 (0.086)	<u>0.830</u> (0.037)
Gemini 2.5*	<u>0.708</u> (0.080)	0.337 (0.078)	<u>0.820</u> (0.041)
WordNet*	0.616 (0.068)	0.378 (0.063)	<u>0.819</u> (0.039)
Handcrafted	<u>0.735</u> (0.129)	<u>0.300</u> (0.100)	<u>0.831</u> (0.038)
ECDF	<u>0.743</u> (0.117)	<u>0.298</u> (0.091)	**0.832** (0.036)
Feature Embeddings	**0.774** (0.104)	**0.263** (0.084)	<u>0.830</u> (0.040)

Table 5: Average metrics for OOD classes *tram*, *cycling*, and *sitting* from NFI_FARED for different hierarchy generation methods. Methods marked with an asterix * are knowledge-driven, while the others are generated using data from the training set. Best results are in bold. Underlined scores indicate models whose performance is not statistically significantly different ($p \geq 0.05$) from the best-performing model, according to a paired t-test over the folds.

to their optimal values.

4.5 Alternative Hierarchies

The choice of hierarchy generation method also has an influence on the performance of Hi-OSCAR. To gauge this influence, we tested a number of different hierarchy generation methods. Three data-driven methods were included, which used HAC as described in Section 2.2 with different input features to generate the hierarchy. The *Feature Embeddings* method uses the features from the embedding space as described in Section 2.2, *ECDF* uses the Empirical Cumulative Distribution Function for each axis, and *Handcrafted* uses handcrafted features such as mean, variance, skew, kurtosis, etc. from the windows of data. All features are calculated using only training samples. The three knowledge-driven hierarchies are generated purely from the class labels. Methods *Chat-GPT 4.5* and *Gemini 2.5* are generated by feeding the list of training classes into the respective LLMs and asking for a hierarchy to be produced. The prompt used is provided in Appendix A.2. The *WordNet* hierarchy is created using the hierarchical relations from the WordNet [27] database. Classes *cycling*, *tram*, and *sitting* from NFI_FARED were used individually as OOD classes and results averaged and displayed in Table 5. The selected classes were chosen for being dissimilar in nature and assessing different aspects of the methods' OOD detection to give an indication of overall performance

The best performing hierarchy in terms of OOD detection was Feature Embeddings, scoring 0.774 and 0.263 for AUROC and detection error, respectively. However, the two other data-driven methods were only marginally worse, and were not significantly different according to a paired t-test. Gemini 2.5 was the best knowledge-driven hierarchy, and was not significantly worse, although the effect size was quite large at -0.8. All hierarchies had similar F1-scores. This indicates that hierarchy choice effects primarily OOD detection in Hi-OSCAR, and has minimal impact on ID performance.

4.6 Ablation Study

To test the contribution of the different components of Hi-OSCAR, experiments were repeated for OOD classes *tram*, *cycling*, and *sitting* ablating over the hierarchy and pretrained weights from Yuan et al. [85]. ID performance was consistent between the ID and OOD experiments, so the ID metrics reported here are reflective of the overall ID performance. The hierarchy was replaced with a single layer of $|K|$ nodes. Consequently, only L_{ID} (1) could be used as the loss function. To ablate over the self-supervised pretrained weights, the feature extractor is randomly initialised and trained from scratch. Results are displayed in Table 6.

Hierarchy	SSL	AUROC ↑	Detection Error ↓	F1 ↑
✗	✗	0.578 (0.110)	0.385 (0.041)	0.658 (0.086)
✓	✗	0.554 (0.138)	0.387 (0.046)	0.686 (0.079)
✗	✓	0.683 (0.087)	0.310 (0.080)	0.833 (0.038)
✓	✓	0.774 (0.104)	0.263 (0.084)	0.830 (0.040)

Table 6: Average metrics for OOD classes *tram*, *cycling*, and *sitting* from NFI_FARED using different components of the Hi-OSCAR architecture. SSL ✗ does not use the pretrained weights and initialises them randomly, and Hierarchy ✗ replaces the hierarchy component of with a single softmax layer. Loss functions are adjusted accordingly.

Training the feature extractor from scratch has the biggest impact on both OOD and ID performance, with results far below both setups employing the pretrained weights. Training from scratch also reduces the quality of the generated hierarchy, and is not able to create a sufficiently representative hierarchy. This hampers OOD performance as a consequence. When combined with pretraining, the hierarchy improves OOD performance by a significant margin compared to using no hierarchy at all. However, ID performance between the hierarchical and non-hierarchical models is equivalent. The primary performance benefit of the hierarchy is in OOD detection, as well as the OOD class approximation enabled by the hierarchy.

4.7 Impact of Window Size

		OOD		ID
Window length	Dataset	AUROC ↑	Detection Error ↓	F1 ↑
5 seconds	NFI_FARED	0.681 (0.115)	0.322 (0.063)	0.727 (0.048)
	OPPORTUNITY	0.523 (0.356)	0.201 (0.144)	0.797 (0.044)
10 seconds	NFI_FARED	0.774 (0.263)	0.263 (0.101)	0.833 (0.032)
	OPPORTUNITY	0.820 (0.048)	0.188 (0.042)	0.828 (0.050)

Table 7: Results of using different window sizes for OOD classes *cycling*, *tram*, and *sitting* from NFI_FARED, and the NULL class from OPPORTUNITY.

Above experiments were all performed using a window size of ten seconds, which is towards the upper end of typical window lengths utilised in HAR. Table 7 compares performance when using instead shorter windows of five seconds, on both NFI_FARED and OPPORTUNITY. For both datasets, ten second windows perform better on the ID and OOD metrics. This is likely due to two factors: firstly, the longer windows provide more context and information, leading to better results, and secondly, the five second model is a smaller size compared to the ten second, also influencing results.

5 Discussion

5.1 Open-set Classification

Effective open-set classification requires both sufficient ID and OOD performance. For ID tasks, Hi-OSCAR outperforms all baselines with the exception of Random Forest on OPPORTUNITY, the smallest tested dataset. While baselines score within 2% of Hi-OSCAR on the datasets PAMAP, OPPORTUNITY, and RealWorld, on the more complex data within NFI_FARED, there is a large gap in performance, with Hi-OSCAR far outperforming baselines. Additionally, Hi-OSCAR has the smallest

performance difference from the smaller datasets to the larger NFI_FARED. This demonstrates a good ability of the model to generalise to different settings.

On OOD tasks, Hi-OSCAR again performs well, beating baselines on average. Additionally, although OpenNet scores equally as well as Hi-OSCAR on OOD tasks, this came at the expense of its ID performance, which is 11% lower than Hi-OSCAR on average, which is of course a major problem for open-set classification. Conversely, kNN matches Hi-OSCAR on ID tasks, but is worse on OOD detection. Our use of a hierarchy allows effective simulation of OOD classes without compromising ID performance, producing a balanced open-set classifier.

For all OOD methods, performance varies between classes, but interestingly, the variations depend on the method. The methods each have unique difficulties for some OOD classes, which are no problem for others. We find that the two best OOD methods, Hi-OSCAR and OpenNet, focus more on different parts of the sensor input to reach conclusions. These different focuses are what give rise to the variability in OOD detection.

5.2 Approximating OOD classes

Use of a hierarchical classifier in our set up provides the additional benefit of assigning more information to samples labelled OOD, and using HAC with Cosine linkage provides a way for these predictions to be evaluated quantitatively. We show that optimal internal predictions could be returned by setting inference stopping criterion $\hat{\lambda} = 0.85$. This on average returned the closest possible node to the true OOD class.

The use of Cumulative Cosine Distance as a measure of prediction quality also helps evaluate overall OOD performance. For example, although for Hi-OSCAR OOD class *running* is weakest, the cosine distances of predictions even when $\hat{\lambda} = 1.00$ are extremely low, meaning that despite predictions being wrong for this class, they are still good approximations of the true value. For many applications of HAR, this can be acceptable, particularly given the overlap between running and similar activities.

It was also demonstrated that different hierarchy generation choices can be used without major loss of performance. Although data-driven hierarchies were overall better due to improved modeling of the inherent physical similarities of activities rather than semantic similarities, Gemini 2.5 was shown to produce an effective hierarchy. This may be useful for a practitioner emphasising semantic interpretation of results. However, adopting such a knowledge-driven method removes the cosine distance metric from the hierarchy, limiting the interpretability of returned internal nodes.

5.3 Dataset NFI_FARED

This work also sees the introduction of NFI_FARED, a new HAR dataset. Compared to existing datasets it has more diverse activities and a large amount of data in terms of length. All tested models perform worst on NFI_FARED ID tasks by a significant margin. The dataset presents an increased challenge that demands more rounded approaches. The similarity between classes requires more fine grain representations to be learned, requiring more robust and adaptable HAR classifiers such as Hi-OSCAR. Additional challenge comes from the data collection procedure. There is deliberate within-class variations such as running and walking up stairs or sitting with and without a backrest. The result is that even individual classes are more difficult to classify by broadening the class definitions to include realistic intra-class diversity.

NFI_FARED is also utilised for OOD classification, a task with limited feasibility using existing datasets. On this aspect there is still room for improvement. The challenge of the classes themselves, combined with the difficulty in distinguishing OOD classes from overlapping ID classes increases the intricacy of OOD detection.

The difficulty in OOD detection is apparent in the variability of OOD performance present in all tested methods. Baselines struggle in some cases where there is significant overlap between ID and

OOD classes. In general machine learning, this phenomenon is known as near-OOD detection, and is typically much more challenging than far-OOD detection, where OOD classes contain obvious domain shifts from ID classes. Some candidates for near-OOD classes could be determined from the data collection procedure. For example, the similarity between *stair up/down* and *escalator up/down*, is obvious, but also classes *kicking* and *punching* could be classed near-OOD. The two activities were collected consecutively during one session with a punching bag. Dividing classes into near- and far-OOD could be a promising avenue of research, but also presents a new set of problems to solve. While existing datasets such as OPPORTUNITY can be used for OOD detection with the NULL class, the lack of information about the NULL samples means possible analysis is limited when compared to NFI_FARED. Overall, there is a considerable deal of further research enabled by NFI_FARED.

6 Conclusions and Future Work

In this work we present our new model Hi-OSCAR for open-set classification on HAR data. Hi-OSCAR provides the best balance between ID and OOD classification, mimicking the real-world requirements of HAR. This is incorporated into the model architecture and training to improve results, and also provides a continuous measure of similarity between classes which is used to evaluate the quality of predictions. To the best of our knowledge, we are also the first to model HAR classes as a hierarchy and utilise Cosine Distance in this way. Additionally, we introduce NFI_FARED, a challenging new HAR dataset. In particular, the variation in similarities between activity classes in the dataset increases the difficulty of OOD detection. In future work, we will look to leverage the internal node information to form novel classes and update the hierarchy, forming an adaptive network.

References

[1] M. A. A. Al-qaness, A. Dahou, M. A. Elaziz, and A. M. Helmi. Multi-ResAtt: Multilevel Residual Network With Attention for Human Activity Recognition Using Wearable Sensors. *IEEE Transactions on Industrial Informatics*, 19(1):144–152, Jan. 2023.

[2] T. T. Alemayoh, J. H. Lee, and S. Okamoto. New Sensor Data Structuring for Deeper Feature Extraction in Human Activity Recognition. *Sensors*, 21(8):2814, Jan. 2021. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.

[3] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz. A Public Domain Dataset for Human Activity Recognition Using Smartphones. *Computational Intelligence*, 2013.

[4] L. Arrotta, G. Civitarese, and C. Bettini. DeXAR: Deep Explainable Sensor-Based Activity Recognition in Smart-Home Environments. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(1):1:1–1:30, Mar. 2022.

[5] L. Arrotta, G. Civitarese, and C. Bettini. Semantic Loss: A New Neuro-Symbolic Approach for Context-Aware Human Activity Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 7(4):147:1–147:29, Jan. 2024.

[6] Y. Asim, M. A. Azam, M. Ehatisham-ul Haq, U. Naeem, and A. Khalid. Context-Aware Human Activity Recognition (CAHAR) in-the-Wild Using Smartphone Accelerometer. *IEEE Sensors Journal*, 20(8):4361–4371, Apr. 2020. Conference Name: IEEE Sensors Journal.

[7] F. Attal, S. Mohammed, M. Dedabrishvili, F. Chamroukhi, L. Oukhellou, and Y. Amirat. Physical Human Activity Recognition Using Wearable Sensors. *Sensors*, 15(12):31314–31338, Dec. 2015. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[8] S. Balli, E. A. Sağbaş, and M. Peker. Human activity recognition from smart watch sensor data using a hybrid of principal component analysis and random forest algorithm. *Measurement and Control*, 52(1-2):37–45, Jan. 2019. Publisher: SAGE Publications Ltd.

[9] A. Bartolome, S. Shah, and T. Prioleau. GlucoMine: A Case for Improving the Use of Wearable Device Data in Diabetes Management. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(3):90:1–90:24, Sept. 2021.

[10] A. Bayat, M. Pomplun, and D. A. Tran. A Study on Human Activity Recognition Using Accelerometer Data from Smartphones. *Procedia Computer Science*, 34:450–457, 2014.

[11] O. Baños, M. Damas, H. Pomares, I. Rojas, M. A. Tóth, and O. Amft. A benchmark dataset to evaluate sensor displacement in activity recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing - UbiComp '12*, page 1026, Pittsburgh, Pennsylvania, 2012. ACM Press.

[12] B. R. Belcher, D. L. Wolff-Hughes, E. E. Dooley, J. Staudenmayer, D. Berrigan, M. S. Eberhardt, and R. P. Troiano. U.S. Population-referenced Percentiles for Wrist-Worn Accelerometer-derived Activity. *Medicine and science in sports and exercise*, 53(11):2455–2464, Nov. 2021.

[13] A. Bendale and T. E. Boult. Towards Open Set Deep Networks. pages 1563–1572, 2016.

[14] M. Berchtold, M. Budde, D. Gordon, H. R. Schmidtke, and M. Beigl. ActiServ: Activity Recognition Service for mobile phones. In *International Symposium on Wearable Computers (ISWC) 2010*, pages 1–8, Oct. 2010. ISSN: 2376-8541.

[15] P. Boyer, D. Burns, and C. Whyne. Out-of-Distribution Detection of Human Activity Recognition with Smartwatch Inertial Sensors. *Sensors*, 21(5):1669, Jan. 2021. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

[16] C. Chatzaki, M. Pediaditis, G. Vavoulas, and M. Tsiknakis. Human Daily Activity and Fall Recognition Using a Smartphone's Acceleration Sensor. In C. Röcker, J. O'Donoghue, M. Ziefle, M. Helfert, and W. Molloy, editors, *Information and Communication Technologies for Ageing Well and e-Health*, pages 100–118, Cham, 2017. Springer International Publishing.

[17] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. d. R. Millán, and D. Roggen. The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recognition Letters*, 34(15):2033–2042, Nov. 2013.

[18] L. Chen, X. Liu, L. Peng, and M. Wu. Deep learning based multimodal complex human activity recognition using wearable devices. *Applied Intelligence*, 51(6):4029–4042, June 2021.

[19] X. Chen, Y. Xiao, Y. Tang, J. Fernandez-Mendoza, and G. Cao. ApneaDetector: Detecting Sleep Apnea with Smartwatches. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(2):59:1–59:22, June 2021.

[20] Y. Chen, W. Cui, Y. Huang, C. Liu, and T. Zhu. Open-Set Sensor Human Activity Recognition Based on Reciprocal Time Series. In Z. Shi, J. Torresen, and S. Yang, editors, *Intelligent Information Processing XII*, pages 101–115, Cham, 2024. Springer Nature Switzerland.

[21] Y. Chen and Y. Xue. A Deep Learning Approach to Human Activity Recognition Based on Single Accelerometer. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1488–1492, Oct. 2015.

[22] J. Cherian, S. Ray, P. Taele, J. I. Koh, and T. Hammond. Exploring the Impact of the NULL Class on In-the-Wild Human Activity Recognition. *Sensors*, 24(12):3898, Jan. 2024. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[23] I. Dirgová Luptáková, M. Kubovčík, and J. Pospíchal. Wearable Sensor-Based Human Activity Recognition with Transformer Model. *Sensors*, 22(5):1911, Jan. 2022. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

[24] A. Djurisic, N. Bozanic, A. Ashok, and R. Liu. Extremely Simple Activation Shaping for Out-of-Distribution Detection, May 2023. arXiv:2209.09858 [cs].

[25] A. Doherty, D. Jackson, N. Hammerla, T. Plötz, P. Olivier, M. H. Granat, T. White, V. T. v. Hees, M. I. Trenell, C. G. Owen, S. J. Preece, R. Gillions, S. Sheard, T. Peakman, S. Brage, and N. J. Wareham. Large Scale Population Assessment of Physical Activity Using Wrist Worn Accelerometers: The UK Biobank Study. *PLOS ONE*, 12(2):e0169649, Feb. 2017. Publisher: Public Library of Science.

[26] X. Du, Z. Wang, M. Cai, and Y. Li. VOS: Learning What You Don't Know by Virtual Outlier Synthesis, May 2022. arXiv:2202.01197 [cs].

[27] C. Fellbaum. WordNet. In R. Poli, M. Healy, and A. Kameas, editors, *Theory and Applications of Ontology: Computer Applications*, pages 231–243. Springer Netherlands, Dordrecht, 2010.

[28] N. Gao, M. S. Rahaman, W. Shao, K. Ji, and F. D. Salim. Individual and Group-wise Classroom Seating Experience: Effects on Student Engagement in Different Courses. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(3):115:1–115:23, Sept. 2022.

[29] N. Gao, W. Shao, M. S. Rahaman, and F. D. Salim. n-Gage: Predicting in-class Emotional, Behavioural and Cognitive Engagement in the Wild. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(3):79:1–79:26, Sept. 2020.

[30] R. Gao, C. Zhao, L. Hong, and Q. Xu. DIFFGUARD: Semantic Mismatch-Guided Out-of-Distribution Detection Using Pre-Trained Diffusion Models. pages 1579–1589, 2023.

[31] M. Hassen and P. K. Chan. Learning a Neural-network-based Representation for Open Set Recognition. In *Proceedings of the 2020 SIAM International Conference on Data Mining (SDM)*, Proceedings, pages 154–162. Society for Industrial and Applied Mathematics, Jan. 2020.

[32] A. M. Helmi, M. A. A. Al-qaness, A. Dahou, and M. Abd Elaziz. Human activity recognition using marine predators algorithm with deep learning. *Future Generation Computer Systems*, 142:340–350, May 2023.

[33] D. Hendrycks and K. Gimpel. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks, Oct. 2018. arXiv:1610.02136 [cs].

[34] D. Hendrycks, M. Mazeika, and T. Dietterich. Deep Anomaly Detection with Outlier Exposure, Jan. 2019. arXiv:1812.04606 [cs].

[35] R. Huang, A. Geng, and Y. Li. On the Importance of Gradients for Detecting Distributional Shifts in the Wild. In *Advances in Neural Information Processing Systems*, volume 34, pages 677–689. Curran Associates, Inc., 2021.

[36] L. P. Jain, W. J. Scheirer, and T. E. Boult. Multi-class Open Set Recognition Using Probability of Inclusion. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 393–409, Cham, 2014. Springer International Publishing.

[37] L. Jennings, M. Sorell, and H. G. Espinosa. Interpreting the location data extracted from the Apple Health database. *Forensic Science International: Digital Investigation*, 44:301504, Mar. 2023.

[38] S. Ji, X. Zheng, and C. Wu. HARGPT: Are LLMs Zero-Shot Human Activity Recognizers?, Mar. 2024. arXiv:2403.02727 [cs].

[39] H. Jia, J. Hu, and W. Hu. SwingNet: Ubiquitous Fine-Grained Swing Tracking Framework via Stochastic Neural Architecture Search and Adversarial Learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(3):106:1–106:21, Sept. 2021.

[40] W. Jung, A. Watson, S. Kuehn, E. Korem, K. Koltermann, M. Sun, S. Wang, Z. Liu, and G. Zhou. LAX-Score: Quantifying Team Performance in Lacrosse and Exploring IMU Features towards Performance Enhancement. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(3):109:1–109:28, Sept. 2021.

[41] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman. The Kinetics Human Action Video Dataset, May 2017. arXiv:1705.06950 [cs].

[42] A. Kilgarriff. Review of WordNet: An Electronic Lexical Database. *Language*, 76(3):706–708, 2000. Publisher: Linguistic Society of America.

[43] M. Kose, O. D. Incel, and C. Ersoy. Online Human Activity Recognition on Smart Phones.

[44] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A large video database for human motion recognition. In *2011 International Conference on Computer Vision*, pages 2556–2563, Nov. 2011. ISSN: 2380-7504.

[45] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *ACM SIGKDD Explorations Newsletter*, 12(2):74–82, Mar. 2011.

[46] J. Li, H. Xu, and Y. Wang. Multiresolution Fusion Convolutional Network for Open Set Human Activity Recognition. *IEEE Internet of Things Journal*, 10(13):11369–11382, July 2023. Conference Name: IEEE Internet of Things Journal.

[47] S. Liang, Y. Li, and R. Srikant. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks, Aug. 2020. arXiv:1706.02690 [cs].

[48] R. Linderman, J. Zhang, N. Inkawhich, H. Li, and Y. Chen. Fine-grain Inference on Out-of-Distribution Data with Hierarchical Classification, Sept. 2022. arXiv:2209.04493 [cs].

[49] W. Liu, X. Wang, J. Owens, and Y. Li. Energy-based Out-of-distribution Detection. In *Advances in Neural Information Processing Systems*, volume 33, pages 21464–21475. Curran Associates, Inc., 2020.

[50] M. Malekzadeh, R. G. Clegg, A. Cavallaro, and H. Haddadi. Protecting Sensory Data against Sensitive Inferences. In *Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems*, W-P2DS'18, pages 1–6, New York, NY, USA, Apr. 2018. Association for Computing Machinery.

[51] S. Mekruksavanich and A. Jitpattanakul. Deep Convolutional Neural Network with RNNs for Complex Activity Recognition Using Wrist-Worn Wearable Sensor Data. *Electronics*, 10(14):1685, Jan. 2021. Number: 14 Publisher: Multidisciplinary Digital Publishing Institute.

[52] S. Mekruksavanich, A. Jitpattanakul, K. Sitthithakerngkiet, P. Youplao, and P. Yupapin. ResNet-SE: Channel Attention-Based Deep Residual Network for Complex Activity Recognition Using Wrist-Worn Wearable Sensors. *IEEE Access*, 10:51142–51154, 2022. Conference Name: IEEE Access.

[53] D. Micucci, M. Mobilio, and P. Napoletano. UniMiB SHAR: A Dataset for Human Activity Recognition Using Acceleration Data from Smartphones. *Applied Sciences*, 7(10):1101, Oct. 2017. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute.

[54] S. Mohsen, A. Elkaseer, and S. G. Scholz. Human Activity Recognition Using K-Nearest Neighbor Machine Learning Algorithm. In S. G. Scholz, R. J. Howlett, and R. Setchi, editors, *Sustainable Design and Manufacturing*, pages 304–313, Singapore, 2022. Springer.

[55] V. S. Murahari and T. Plötz. On attention models for human activity recognition. In *Proceedings of the 2018 ACM International Symposium on Wearable Computers*, pages 100–103, Singapore Singapore, Oct. 2018. ACM.

[56] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. Do Deep Generative Models Know What They Don't Know?, Feb. 2019. arXiv:1810.09136 [stat].

[57] J. Nie, Y. Luo, S. Ye, Y. Zhang, X. Tian, and Z. Fang. Out-of-Distribution Detection with Virtual Outlier Smoothing. *International Journal of Computer Vision*, 133(2):724–741, Feb. 2025.

[58] N. R. Nurwulan and G. Selamaj. Human daily activities recognition using decision tree. *Journal of Physics: Conference Series*, 1833(1):012039, Mar. 2021. Publisher: IOP Publishing.

[59] T. Okita, K. Ukita, K. Matsuishi, M. Kagiyama, K. Hirata, and A. Miyazaki. Towards LLMs for Sensor Data: Multi-Task Self-Supervised Learning. In *Adjunct Proceedings of the 2023 ACM International Joint Conference on Pervasive and Ubiquitous Computing & the 2023 ACM International Symposium on Wearable Computing*, pages 499–504, Cancun, Quintana Roo Mexico, Oct. 2023. ACM.

[60] F. J. Ordóñez and D. Roggen. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors*, 16(1):115, Jan. 2016. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[61] P. Perera, V. I. Morariu, R. Jain, V. Manjunatha, C. Wigington, V. Ordonez, and V. M. Patel. Generative-Discriminative Feature Representations for Open-Set Recognition. pages 11814–11823, 2020.

[62] C. Randell and H. Muller. Context awareness by analysing accelerometer data. In *Digest of Papers. Fourth International Symposium on Wearable Computers*, pages 175–176, Oct. 2000.

[63] A. Reiss and D. Stricker. Introducing a New Benchmarked Dataset for Activity Monitoring. In *2012 16th International Symposium on Wearable Computers*, pages 108–109, June 2012. ISSN: 2376-8541.

[64] M. Shoaib, S. Bosch, O. D. Incel, H. Scholten, and P. J. M. Havinga. Fusion of Smartphone Motion Sensors for Physical Activity Recognition. *Sensors*, 14(6):10146–10176, June 2014. Number: 6 Publisher: Multidisciplinary Digital Publishing Institute.

[65] M. Shoaib, S. Bosch, O. D. Incel, H. Scholten, and P. J. M. Havinga. Complex Human Activity Recognition Using Smartphone and Wrist-Worn Motion Sensors. *Sensors*, 16(4):426, Apr. 2016. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[66] G. A. Sigurdsson, A. Gupta, C. Schmid, A. Farhadi, and K. Alahari. Charades-Ego: A Large-Scale Dataset of Paired Third and First Person Videos, Apr. 2018. arXiv:1804.09626 [cs].

[67] S. P. Singh, M. K. Sharma, A. Lay-Ekuakille, D. Gangwar, and S. Gupta. Deep ConvLSTM With Self-Attention for Human Activity Decoding Using Wearable Sensors. *IEEE Sensors Journal*, 21(6):8575–8582, Mar. 2021.

[68] A. Stisen, H. Blunck, S. Bhattacharya, T. S. Prentow, M. B. Kjærgaard, A. Dey, T. Sonne, and M. M. Jensen. Smart Devices are Different: Assessing and MitigatingMobile Sensing Heterogeneities for Activity Recognition. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 127–140, Seoul South Korea, Nov. 2015. ACM.

[69] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3319–3328. PMLR, July 2017. ISSN: 2640-3498.

[70] T. Sztyler and H. Stuckenschmidt. On-body localization of wearable devices: An investigation of position-aware activity recognition. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9, Mar. 2016.

[71] Y. Tang, L. Zhang, Q. Teng, F. Min, and A. Song. Triple Cross-Domain Attention on Human Activity Recognition Using Wearable Sensors. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(5):1167–1176, Oct. 2022. Conference Name: IEEE Transactions on Emerging Topics in Computational Intelligence.

[72] L. Tao, X. Du, X. Zhu, and Y. Li. Non-Parametric Outlier Synthesis, Mar. 2023. arXiv:2303.02966 [cs].

[73] J. Tayyub, M. Hawasly, D. C. Hogg, and A. G. Cohn. Learning Hierarchical Models of Complex Daily Activities from Annotated Videos. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1633–1641, Mar. 2018.

[74] M. T. H. Tonmoy, S. Mahmud, A. K. M. M. Rahman, M. A. Amin, and A. A. Ali. Hierarchical Self Attention Based Autoencoder for Open-Set Human Activity Recognition, Mar. 2021. arXiv:2103.04279.

[75] D. N. Tran and D. D. Phan. Human Activities Recognition in Android Smartphone Using Support Vector Machine. In *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pages 64–68, Bangkok, Thailand, Jan. 2016. IEEE.

[76] J. P. van Zandwijk and A. Boztas. The iPhone Health App from a forensic perspective: can steps and distances registered during walking and running be used as digital evidence? *Digital investigation*, 28:S126–S133, 2019. Publisher: Elsevier.

[77] J. P. van Zandwijk and A. Boztas. The phone reveals your motion: Digital traces of walking, driving and other movements on iPhones. *Forensic Science International: Digital Investigation*, 37:301170, June 2021.

[78] A. Wan, L. Dunlap, D. Ho, J. Yin, S. Lee, H. Jin, S. Petryk, S. A. Bargal, and J. E. Gonzalez. NBDT: Neural-Backed Decision Trees, Jan. 2021. arXiv:2004.00221.

[79] C. Wang, Y. Gao, A. Mathur, A. C. De C. Williams, N. D. Lane, and N. Bianchi-Berthouze. Leveraging Activity Recognition to Enable Protective Behavior Detection in Continuous Data. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(2):81:1–81:27, June 2021.

[80] L. Wang and R. Liu. Human Activity Recognition Based on Wearable Sensor Using Hierarchical Deep LSTM Networks. *Circuits, Systems, and Signal Processing*, 39(2):837–856, Feb. 2020.

[81] Q. Wang, J. Ye, F. Liu, Q. Dai, M. Kalander, T. Liu, J. Hao, and B. Han. Out-of-distribution Detection with Implicit Outlier Transformation, Mar. 2023. arXiv:2303.05033 [cs].

[82] H. Xu, J. Li, H. Yuan, Q. Liu, S. Fan, T. Li, and X. Sun. Human Activity Recognition Based on Gramian Angular Field and Deep Convolutional Neural Network. *IEEE Access*, 8:199393–199405, 2020. Conference Name: IEEE Access.

[83] H. Xu, P. Zhou, R. Tan, M. Li, and G. Shen. LIMU-BERT: Unleashing the Potential of Unlabeled Data for IMU Sensing Applications. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, pages 220–233, Coimbra Portugal, Nov. 2021. ACM.

[84] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher. DeepSense: A Unified Deep Learning Framework for Time-Series Mobile Sensing Data Processing. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 351–360, Republic and Canton of Geneva, CHE, Apr. 2017. International World Wide Web Conferences Steering Committee.

[85] H. Yuan, S. Chan, A. P. Creagh, C. Tong, D. A. Clifton, and A. Doherty. Self-supervised Learning for Human Activity Recognition Using 700,000 Person-days of Wearable Data, June 2022. arXiv:2206.02909 [cs, eess] version: 1.

[86] Y. Zhou. Rethinking Reconstruction Autoencoder-Based Out-of-Distribution Detection. pages 7379–7387, 2022.

[87] J. Zhu, R. San-Segundo, and J. M. Pardo. Feature extraction for robust physical activity recognition. *Human-centric Computing and Information Sciences*, 7(1):16, June 2017.

A Appendix

A.1 Baseline Hyperparameters

Refer to Table 8

Model	Optimiser	Learning Rate	Details
CNNBiGRU	Adam	1e-4	num_filters: 64, filter_size: 8, n_units_gru: 128, n_layers_gru: 2
Transformer	Adam	1e-3	embedding_dim: 128, num_heads: 8, num_layers: 3, label_smoothing: 0.1
ResNet-SE	Adam	1e-4	kernel_size: 5, conv_stride: 1, conv_num_filters: 64, res_num_filters: 32, layer_size: 128
CNNLSTM-Att	RMSProp	1e-3	hidden_size: 128, dropout: 0.6, attention_dropout: 0.7, num_layers: 2, num_filters: 64
MLP	Adam	1e-4	layer1_size: 128, layer2_size: 256, layer3_size: 512, layer4_size: 1024
Random Forest	n/a	n/a	mean, std, range, med abs dev, kurt, skew, error norm mean. n_estimators=3000, imblearn default

Table 8: Hyperparameters of baselines.

A.2 LLM Prompt

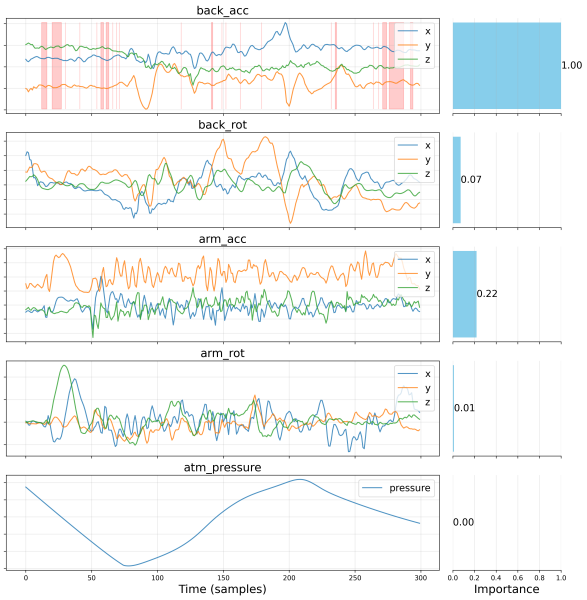
Prompt used to create hierarchies with LLMs Chat-GPT 4.5 and Gemini 2.5: "*[list of activities]*: This is a list of activities that I have in my dataset. Please arrange them in a sensible hierarchy. I want them organised so all activities are part of the same hierarchy i.e. there is a root node that is an ancestor to all activities"

A.3 Multiple classes in OOD set

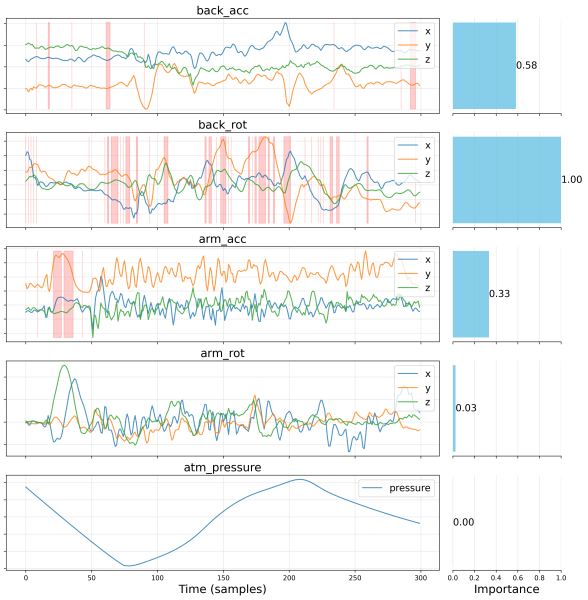
Refer to Table 9

		OOD		ID
Model	OOD set	AUROC ↑	Detection Error ↓	F1 ↑
OpenNet [31]	(sitting, bus)	0.625 (0.132)	0.387 (0.095)	0.739 (0.076)
	(stair.down, dragging, throwing)	0.498 (0.171)	0.419 (0.071)	0.662 (0.054)
	(sitting, running, bus)	**0.668** (0.113)	**0.360** (0.079)	0.717 (0.061)
	(cycling, kicking, tram)	0.730 (0.087)	0.304 (0.056)	0.755 (0.096)
	(standing, walking, elevator.up)	0.632 (0.091)	0.384 (0.049)	0.710 (0.060)
	(running, elevator.down, throwing, bus)	0.628 (0.119)	0.371 (0.066)	0.662 (0.096)
	(stair.up, escalator.down, escalator.up, car)	**0.715** (0.058)	**0.320** (0.045)	0.736 (0.059)
VAE	(sitting, bus)	0.584 (0.098)	0.396 (0.058)	0.609 (0.038)
	(stair.down, dragging, throwing)	0.501 (0.099)	0.396 (0.068)	0.575 (0.036)
	(sitting, running, bus)	0.549 (0.072)	0.426 (0.041)	0.533 (0.094)
	(cycling, kicking, tram)	0.505 (0.098)	0.427 (0.039)	0.612 (0.041)
	(standing, walking, elevator.up)	0.595 (0.063)	0.402 (0.050)	0.635 (0.029)
	(running, elevator.down, throwing, bus)	0.493 (0.133)	0.424 (0.061)	0.570 (0.053)
	(stair.up, escalator.down, escalator.up, car)	0.430 (0.119)	0.472 (0.032)	0.634 (0.026)
kNN	(sitting, bus)	0.355 (0.093)	0.472 (0.059)	0.837 (0.060)
	(stair.down, dragging, throwing)	**0.844** (0.048)	**0.217** (0.047)	0.809 (0.051)
	(sitting, running, bus)	0.510 (0.134)	0.404 (0.059)	**0.837** (0.036)
	(cycling, kicking, tram)	0.571 (0.066)	0.398 (0.045)	**0.843** (0.072)
	(standing, walking, elevator.up)	0.468 (0.072)	0.469 (0.024)	0.830 (0.038)
	(running, elevator.down, throwing, bus)	0.613 (0.055)	0.379 (0.024)	0.802 (0.081)
	(stair.up, escalator.down, escalator.up, car)	0.558 (0.041)	0.419 (0.044)	**0.858** (0.043)
Hi-OSCAR	(sitting, bus)	**0.717** (0.089)	**0.313** (0.066)	**0.841** (0.052)
	(stair.down, dragging, throwing)	0.749 (0.087)	0.303 (0.078)	**0.837** (0.043)
	(sitting, running, bus)	0.608 (0.066)	0.397 (0.041)	**0.848** (0.056)
	(cycling, kicking, tram)	**0.766** (0.077)	**0.303** (0.060)	0.837 (0.030)
	(standing, walking, elevator.up)	**0.829** (0.081)	**0.232** (0.069)	**0.854** (0.039)
	(running, elevator.down, throwing, bus)	**0.662** (0.081)	**0.358** (0.052)	**0.844** (0.039)
	(stair.up, escalator.down, escalator.up, car)	0.673 (0.085)	0.359 (0.052)	**0.870** (0.038)

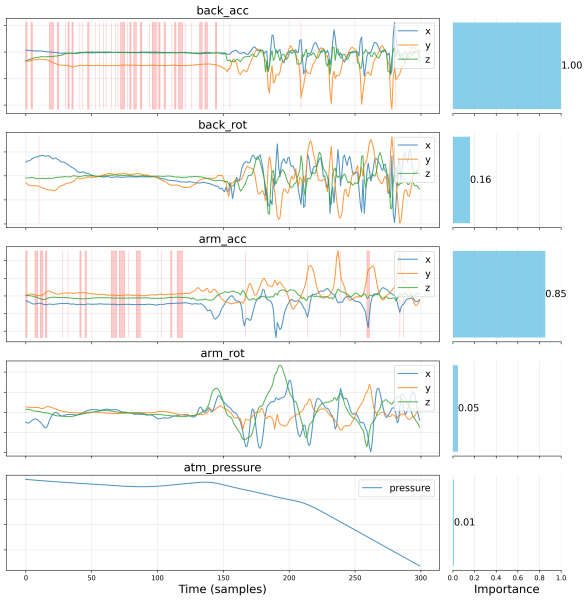
Table 9: OOD results when multiple randomly chosen classes are included in the OOD set. Best results are in bold. Underlined scores indicate models whose performance is not statistically significantly different ($p \geq 0.05$) from the best-performing model, according to a paired t-test over the folds.



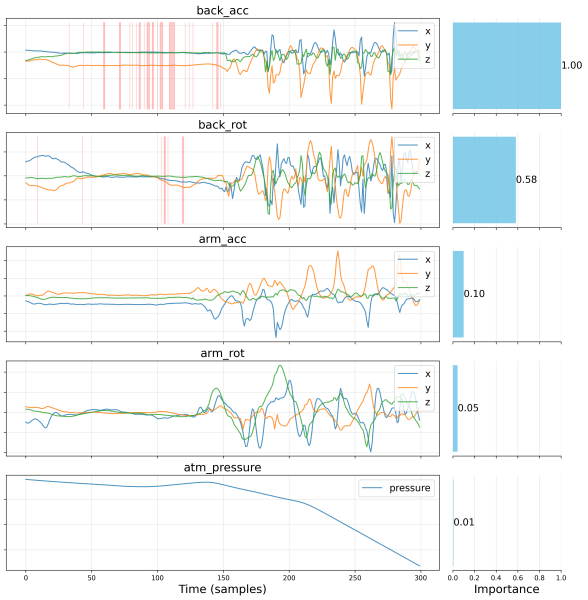
(a) Hi-OSCAR: *cycling*



(b) OpenNet: *cycling*

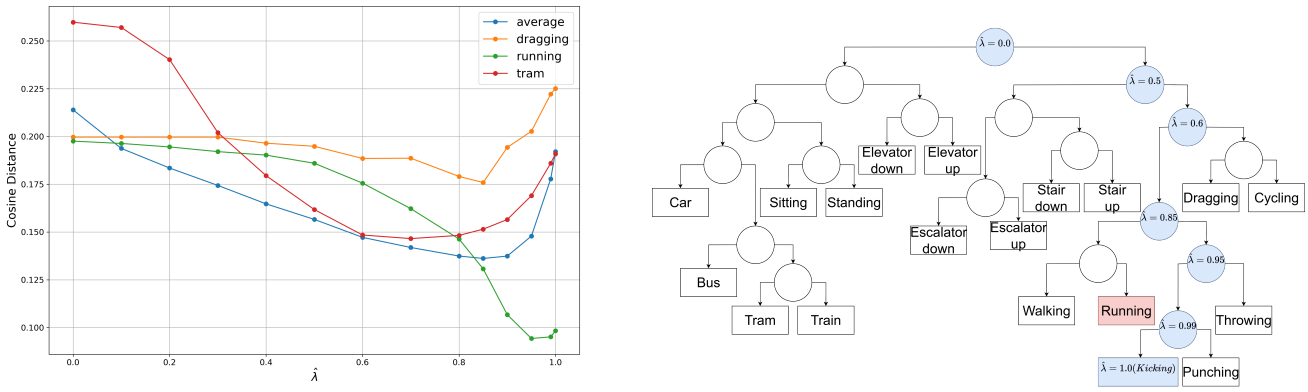


(c) Hi-OSCAR: *running*



(d) OpenNet: *running*

Figure 7: Importance of sections of the input sensor data for the output of Hi-OSCAR vs. OpenNet for classes *cycling* and *running*, calculated using XAI technique Integrated Gradients. Most important sections of the timeseries highlighted in red, and overall sensor importance is shown in the bar graphs to the right. Hi-OSCAR emphasises the back acceleration and arm acceleration sensors primarily to produce output, while OpenNet emphasises sensors back acceleration and back rotation.



(a) (b)

Figure 8: (a) Cumulative Cosine Distance of predicted node from true node position for different OOD classes when varying threshold $\hat{\lambda}$. (b) Example of how predictions (marked in blue) change with $\hat{\lambda}$. The hierarchy was computed using all classes and OOD class *running* was removed during training. Its position in the original hierarchy is marked in red.