International Journal of Science and Engineering
ISSN (Online): 2454-2016
Volume 11 Issue 02 June 2025
DOI: https://doi.org/10.53555/ym736w23

UNSUPERVISED EXPLORATION OF HIERARCHICAL STRUCTURE IN HUMAN ACTIVITY RECOGNITION DATA

Chayse Monteen[1*], Aakash Dhananjay Shanbhag[2]

[1]BTech Computer Science Engineering Vellore Institute of Technology Vellore, India chaysemonteen2@gmail.com*
[2]CA, aakashshanbhag@gmail.com

Abstract*—Human Activity Recognition (HAR) datasets contain complex patterns that supervised models exploit with labeled training, but it remains unclear what latent structure exists in the data itself. This paper presents an unsupervised exploratory analysis of a smartphone sensor HAR dataset to uncover inherent activity groupings without using activity labels. We apply a range of clustering algorithms (k-means, Gaussian mixture, hierarchical agglomerative, density-based HDBSCAN, spectral clustering) and dimensionality reduction methods (Principal Component Analysis – PCA, t-distributed Stochastic Neighbor Embedding – t-SNE, Uniform Manifold Approximation and Projection – UMAP, and a feed-forward autoencoder) to identify natural clusters of sensor feature vectors. Quantitatively, we evaluate clustering quality using internal metrics (silhouette coefficient) and external metrics against true labels (Adjusted Rand Index – ARI, and Normalized Mutual Information – NMI). The results reveal a dominant two-cluster division separating static postures from dynamic movements, with finer sub-clusters roughly corresponding to the six known activities when clustering is applied hierarchically. UMAP non-linear embedding dramatically improved cluster separability and alignment with classes, outperforming PCA. Analyzing feature importance in each cluster showed that features related to body orientation and acceleration dynamics differentiate activities. These findings demonstrate that unsupervised learning can automatically discover meaningful activity groupings (e.g. distinguishing stationary vs. moving behaviors) and key distinguishing sensor features, without any labels. The study provides insights into intrinsic HAR data structure, which can inform feature design and hierarchical modeling in future activity recognition systems.*

Keywords— Human Activity Recognition (HAR); Unsupervised Clustering; Dimensionality Reduction; Wearable Sensors; Feature Analysis.

INTRODUCTION

Human Activity Recognition (HAR) involves interpreting sensor signals from smartphones or wearable devices to understand what a person is doing. Most existing approaches are supervised, relying on labeled datasets and engineered features to build classifiers. However, labels are expensive to obtain, and the large number of sensor features—often running into hundreds—make the data both redundant and difficult to interpret. This raises a natural question: *what intrinsic patterns exist in the raw data itself, without any labels?*

The motivation for this work is to uncover such inherent structures. Human activity data are high-dimensional, reflecting many interacting sensor channels and statistical descriptors. Looking at them directly is overwhelming—similar to trying to make sense of hundreds of moving joints at once. The challenge is that meaningful groupings may be present but hidden in this high-dimensional space.

To address this, we adopt an unsupervised approach. The flow of our study begins with feature reduction: removing duplicates, reducing correlation, and applying techniques such as **Principal Component Analysis (PCA)** [1], **t-distributed Stochastic Neighbor Embedding (t-SNE)** [2], and **Uniform Manifold Approximation and Projection (UMAP)** [3] as well as an autoencoder to compress the data into a more compact and informative representation. This step reduces redundancy while retaining essential signal variation. On these reduced representations, we apply a variety of clustering algorithms—including k-means, Gaussian mixture models, hierarchical agglomerative clustering, density-based methods, and spectral clustering—to see whether the latent groupings emerge more clearly.

The overall process is therefore: start from the full feature set, perform careful reduction, explore multiple dimensionality reduction strategies, and then apply clustering to uncover structure. Finally, we briefly assess how well the discovered clusters correspond to intuitive categories of activities in the **UCI Human Activity Recognition dataset** [4], but our primary focus is on the exploratory discovery of intrinsic patterns rather than on classification accuracy.

I. LITERATURE REVIEW

Unsupervised Deep Clustering for HAR (Amrani *et al.*, 2022)

Amrani *et al.* introduced **Deep Inertial Sensory Clustering (DISC)**, an unsupervised deep learning method that combines a ConvGRU-based autoencoder with a clustering objective to group inertial sensor signals [5]. The encoder learns compact spatio-temporal features, decoders reconstruct past and predict future sequences, and the latent representations are clustered. Evaluated on three public HAR datasets (including UCI Smartphone), DISC outperformed four state-of-the-art methods in clustering accuracy and normalized mutual information. Its main advantage is the ability to learn temporal and spatial dependencies directly from unlabeled data, avoiding hand-crafted features and enabling more cohesive clusters. However, DISC is computationally heavy, requires careful tuning of architecture and latent size, and assumes a known number of clusters—limits for a fully unsupervised setting. Compared to our feature-reduction plus clustering pipeline, DISC provides stronger embeddings but at the cost of higher complexity, while our approach is more lightweight and interpretable.

DCAM-Net for Smartphone HAR (Xu *et al.*, 2025)

Xu *et al.* developed **DCAM-Net (DeepConvAttentionMLPNet)**, a supervised deep model integrating multi-scale CNNs, residual connections, attention modules, and MLPs for smartphone-based HAR [8]. Using the standard 561-feature UCI dataset (30 subjects), it reached **99.03% accuracy** with five-fold cross-validation, reducing misclassification by ~8.5% compared to static sensor fusion. With 2.9M parameters (62% fewer than typical CNN–LSTM baselines), it balances accuracy and efficiency. Its strengths are exceptional accuracy, robust attention-based sensor fusion, and parameter efficiency that supports near real-time use. Limitations include validation on only one dataset, heavy computational demand for on-device deployment, and dependence on labeled data, restricting unsupervised discovery. Unlike our work, which clusters unlabeled data via dimensionality reduction and simple algorithms, DCAM-Net excels at classification of known classes. Our approach sacrifices accuracy but enables flexible exploration of data without labels, complementing deep supervised models.

Deep ConvLSTM (Ordóñez & Roggen, 2016)

Ordóñez and Roggen's **DeepConvLSTM** combined convolutional layers with LSTMs to jointly model local features and temporal dependencies, achieving state-of-the-art supervised HAR on datasets like OPPORTUNITY and Skoda [6]. It improved recognition by 4–9% over CNN-only baselines and eliminated the need for manual features. Its advantages are robust end-to-end learning of spatial and temporal patterns and high throughput (reported at 900× real-time with GPU). However, the model is data-hungry, with >5M parameters, requiring extensive labeled training and making deployment difficult on resource-limited devices [7]. Unlike our unsupervised method, which identifies latent groupings without labels, DeepConvLSTM depends on predefined activity classes. While it achieves superior classification accuracy, our approach is better suited for exploratory analysis when annotations are unavailable.

II. METHODOLOGY

Dataset and Feature Processing

We use the UCI Human Activity Recognition (HAR) dataset, derived from smartphone accelerometer and gyroscope

signals. The dataset contains 10,299 samples, each labeled with one of six activities: LAYING, SITTING, STANDING, WALKING, WALKING_UPSTAIRS, or WALKING_DOWNSTAIRS. Data were collected from multiple subjects performing these activities, segmented into 2.56-second fixed windows (50 Hz sampling), and transformed into 561 engineered features per sample.These features summarize sensor dynamics in both time (t) and frequency (f) domains. For example:
t BodyAcc-mean()-X: mean of body acceleration along the X-axis (time domain).
tBodyAcc-std()-Y: standard deviation of body acceleration along the Y-axis.
tBodyAcc-mad()-Z: median absolute deviation of acceleration along the Z-axis.
tBodyGyro-mean()-X: mean angular velocity from gyroscope (X-axis).
tBodyAccJerk-mean()-Z: mean jerk (derivative of acceleration) on Z-axis.
tBodyGyroJerk-correlation()-X,Z: correlation between gyroscope jerk signals on X and Z axes.
fBodyBodyGyroJerkMag-maxInds: frequency index of the dominant gyroscope jerk magnitude (FFT domain).

Broadly, the feature set includes: Basic statistics (mean, std, max, min, median absolute deviation),Shape descriptors (skewness, kurtosis, entropy), Temporal dynamics (autoregressive coefficients, jerk signals), Cross-axis measures (correlation between axes), and Spectral features (FFT magnitudes, band energy, mean frequency, dominant frequency index). Together, these features capture both posture-related orientation (gravity and static acceleration) and motion dynamics (jerk, frequency signatures of steps). Prior to clustering, we performed feature cleaning and preprocessing to remove redundant or problematic features and to standardize the data:

Duplicate/Constant Features: We detected and dropped 21 duplicate feature columns (features with identical values across all samples). No features were constant or near-constant, so none were removed for low variance.

Highly Correlated Features: To reduce dimensionality and multicollinearity, we dropped features with very high pairwise correlation (absolute Pearson $\rho > 0.98$). This eliminated 194 features that were essentially linear combinations of others, retaining a more independent feature set.

After cleaning, the feature count was reduced from the original 561 to 346 informative features. We verified that no missing values (NaNs) were present initially, and none were introduced during preprocessing. All features were then standardized (scaled to zero mean and unit variance) to ensure comparability.

Data shape: The final processed feature matrix has shape (10,299 samples × 346 features). Each sample also has an associated subject ID (which we do not use in clustering) and an activity label (used only for evaluating clustering results, not provided to the clustering algorithms).

Table 1 summarizes the dataset composition by activity label after preprocessing. The class distribution is fairly balanced, with roughly 1.4k–1.9k samples per activity. This balance is useful for evaluation: clusters that align well with activities will not be dominated by one extremely large class.

Table 1. Distribution of activity classes in the dataset (total samples = 10,299).

Activity	Count of Samples
LAYING	1,944
SITTING	1,777
STANDING	1,906
WALKING	1,722
WALKING_UPSTAIRS	1,544
WALKING_DOWNSTAIRS	1,406
Total	10,299

Clustering Algorithms and Dimensionality Reduction
We applied a broad set of clustering techniques to the processed feature vectors to identify inherent groupings:
Partitioning (Centroid-based) Methods: *k-means* and *Mini-Batch k-means (MBKMeans)* [9] clustering were run for various fixed numbers of clusters *k*. We primarily examined k = 4 through k = 12 clusters to see if the natural grouping matches or splits the 6 known classes.
Model-based Probabilistic Clustering: *Gaussian Mixture Models (GMM)* [10] were used with various component counts (similar k values as k-means), allowing soft clustering and non-spherical clusters.
Hierarchical Clustering: We used agglomerative clustering with Ward's linkage [11] (denoted *Agglo-Ward*), specifying cluster counts in a similar range.
Density-Based Clustering: We employed *HDBSCAN* [12] (Hierarchical DBSCAN) which determines the number of clusters automatically based on density (and can label some points as noise). HDBSCAN has parameters min_samples and min_cluster_size which we tuned in experiments; it does not require choosing *k*.
Spectral Clustering: We performed spectral clustering [13] using the nearest-neighbors graph (denoted *Spectral*), with a specified number of clusters *k*. This method can capture non-convex cluster structures by embedding data into a

graph Laplacian spectral space.

In addition to clustering in the original 346-dimensional feature space, we explored *dimensionality reduction* to see if a lower-dimensional representation could expose clearer clusters:

Principal Component Analysis (PCA): We used PCA to project the data into lower dimensions (e.g., top 2 principal components for visualization, or more components for clustering). Clustering was then run on these PCA-transformed features.

t-distributed Stochastic Neighbor Embedding (t-SNE): t-SNE (with perplexity and learning rate tuned) was used mainly for visualization of cluster tendencies in 2D, since it is non-linear but not ideal for clustering directly (due to no inverse transform and stochastic nature).

Uniform Manifold Approximation and Projection (UMAP): UMAP, a non-linear manifold learning technique, was applied to reduce the data to 2 dimensions (and also to 10 dimensions in some tests). UMAP has parameters (n_neighbors, min_dist) that we varied; it tends to preserve local and some global structure and can create a more clustering-friendly embedding.

Autoencoder Latent Space: We trained a simple feed-forward autoencoder [14] to compress the 346 features into a 32-dimensional latent representation. The autoencoder consists of fully-connected layers and was trained to minimize reconstruction error. After training, we obtained 32-D latent vectors for all samples and applied clustering algorithms in this latent space. The motivation is that the autoencoder might learn a non-linear compression that retains the important structure of the data while removing noise.
All clustering algorithms were run on the standardized feature data (or its reduced forms). For methods requiring *k*, we often examined *k = 6* specifically (to compare with the six known activities) as well as other values to see if the algorithm's optimal clustering diverged from six. We used both internal and external validation: the silhouette coefficient (ranges –1 to 1, indicating how well-separated clusters are, with higher values better), the Adjusted Rand Index (ARI) (ranges 0 to 1, measuring how closely the clustering agrees with ground truth labels after chance correction), and the Normalized Mutual Information (NMI) (ranges 0 to 1, capturing how much information is shared between clustering assignments and true labels). Importantly, the ground-truth labels were never provided to the clustering algorithms themselves, only used afterward for evaluation.

model	clust ers	silhouette	ARI	NMI
HDBSCAN	**3**	**0.40862**	0.21846	0.3526
KMeans(k=4)	4	0.16361	0.29232	0.4638
KMeans(k=5)	5	0.14173	0.28611	0.4557
KMeans(k=6)	6	0.11622	0.29938	0.4613
KMeans(k=8)	8	0.10724	0.31052	0.4668
AggloWard(k= 6)	6	0.10631	0.32636	0.4839
KMeans(k=7)	7	0.10494	0.31787	0.4757

III. Experiments and Results

Quantitative Results

We implemented the above methods in Python with scikit-learn and frameworks like PyTorch and TensorFlow. For k-means and GMM, multiple random initializations were used to avoid unlucky initial seeds. For spectral clustering, we used 10 nearest neighbors for the affinity matrix. UMAP was run with a variety of n_neighbors (e.g., 15, 50) and min_dist (0.0 to 0.5) settings; the primary setting used was n_neighbors=15, min_dist=0.1, n_components=2 for visualization, and also n_components=2 or higher for feeding into clustering. The autoencoder was a shallow network (input → 128 → 64 → 32 dimension encoding, then symmetric decoding), trained for 500 epochs which was sufficient to reach low reconstruction error.

In total, we conducted experiments clustering (A) directly on the 346-D features, (B) on UMAP-reduced features, and (C) on autoencoder latent features. Additionally, a hierarchical two-level clustering approach was tested: first cluster broadly (e.g., HDBSCAN to get 2 clusters), then cluster each resulting cluster separately to see if they subdivide into the actual activities. Below, we present the results of these experiments.

Clustering in Original Feature Space (346-D)
We first applied clustering algorithms *without* any dimensionality reduction to the full 346-dimensional feature vectors. This provides a baseline for how separable the activities are in the original feature space. Table 2 highlights selected clustering outcomes in the raw feature space, including HDBSCAN and several algorithms set to find 6 clusters.

Table 2. Clustering performance on raw 346-D feature space (selected algorithms).
(Silhouette measures cluster cohesion/separation; ARI and NMI are relative to true activity labels.)
In the raw feature space, **HDBSCAN** naturally found 3 clusters as the optimal density-based grouping. These 3 roughly corresponded to a large cluster of static activities and two clusters splitting the dynamic activities (as we analyze later). This yielded the highest silhouette (0.409) among all methods, indicating well-separated broad clusters, but a relatively low ARI (0.218) since 3 clusters cannot finely match 6 actual classes. In contrast, forcing algorithms to find 6 clusters tended to produce much lower silhouette scores (~0.08–0.16), suggesting that the six activity classes are not cleanly separable in the original feature space. Notably, **spectral clustering** with k=6 achieved the highest alignment with true labels (ARI ≈ 0.479, NMI ≈ 0.616) among the raw-space clustering methods, meaning it best captured the actual class distinctions. However, its silhouette was very low (0.083), indicating those 6 spectral clusters were quite overlapping in feature space (poor cohesion). Other methods with k=6 like k-means and GMM had moderate NMI in the 0.43–0.48 range but also low silhouette (~0.10). **Agglomerative (Ward)** clustering at k=6 was similar (ARI 0.33). We observed a general trend: in high dimensions, clustering algorithms struggled to isolate all six activities distinctly, likely due to class overlap and noisy features – except HDBSCAN, which opted to merge some classes and found a higher-level split with better separation. This indicates that the inherent structure of the data, without embedding, is dominated by a broad grouping rather than six perfectly distinct clusters.

Clustering with Non-Linear Embeddings (UMAP)
Next, we evaluated clustering on a lower-dimensional manifold learned by UMAP. We found that applying UMAP to reduce dimensionality significantly improved clustering separation. In particular, projecting the data into 2-D with UMAP and then clustering yielded much better-defined clusters. Using UMAP (2 components) followed by HDBSCAN, for example, resulted in only 2 clusters, but with an excellent silhouette score (~0.72) and much higher ARI (~0.49) and NMI (~0.70) than any clustering directly on raw features. This two-cluster solution corresponded to a sensible division of the data: essentially splitting the samples into **static vs. dynamic activity clusters** (as we discuss later). The high silhouette (~0.72) indicates these two clusters were very well separated in the UMAP space.

When we fixed the number of clusters at 6 (to match the number of known activities) and applied various algorithms on the UMAP-reduced 2D data, we saw substantial performance improvements over clustering in the original space. Table 3 compares clustering algorithms on the 2-D UMAP embedding (with $k = 6$ for those that require it). For reference, HDBSCAN's 2-cluster result is also shown.

Table 3. Clustering performance after UMAP dimensionality reduction (2-D embedding).

Algorithm	Silhouette	ARI	NMI
HDBSCAN	**0.722353**	**0.489704**	**0.700345**
KMeans	0.493042	0.538700	0.666284
MiniBatchK Means	0.418998	0.512631	0.640615
Agglomerative	0.403749	0.546140	0.666551
GMM	0.389912	**0.551343**	**0.671200**
DBSCAN	0.064090	0.320357	0.536286

On the UMAP embedding, **k-means, agglomerative, and GMM** all showed ARI in the ~0.54–0.55 range and NMI ~0.67, notably higher than the best ~0.48 ARI achieved in raw space. The silhouette scores for these 6-cluster solutions (0.39–0.49) were still lower than HDBSCAN's broad 2-cluster solution (0.72), indicating that when forcing six clusters, some clusters in UMAP space are closer together – but overall the separation is much improved compared to raw features. Among these, GMM obtained the highest ARI (~0.551) and NMI (~0.671) for k=6, slightly better than k-means or agglomerative on this embedding. This suggests that in the UMAP space, class clusters might have shapes that a Gaussian mixture models slightly better. **Spectral clustering** on UMAP was an outlier – it performed poorly (ARI 0.18), possibly because the 2D embedding already linearized the structure and the spectral method with k=6 might have difficulty since the inherent optimal cluster count in 2D was really 2. We also attempted DBSCAN on the UMAP result; it tended to either merge into 2–3 clusters or fragment into many micro-clusters depending on ε, and did not achieve high alignment (shown is one run with 6 clusters, ARI 0.32, NMI 0.54).

These results illustrate that **non-linear manifold learning (UMAP)** preserved the cluster structure of activities far better than linear PCA in this case (note: PCA results, not shown in full, yielded ARI only up to ~0.42 for k=6 and much lower silhouette ~0.25). UMAP created a representation where distances between points more closely reflected true activity differences (e.g., all static points clustered together, separated from dynamic points), thereby greatly aiding clustering. The best unsupervised result we obtained was with UMAP + clustering: HDBSCAN on UMAP yielded two

very pure clusters (static vs dynamic split) with silhouette ~0.72 and NMI ~0.70, and when aiming for six clusters, a combination like UMAP + GMM or Agglomerative achieved ARI ≈ 0.55, NMI ≈ 0.67. This is a significant improvement over direct clustering on original features (ARI ~0.30–0.48). It indicates that much of the class structure was nonlinear and UMAP was able to untangle it.

Clustering in Autoencoder Latent Space
To obtain a compact latent representation of the 346-dimensional cleaned feature space, we trained a **fully-connected autoencoder (AE)**. The AE compresses inputs into a 32-dimensional latent space and reconstructs them back to the original feature size, learning a smooth, denoised manifold of the HAR data.

Architecture
Encoder: Input (346) → Dense (512, ReLU) → Dense (128, ReLU) → Dense (32, latent).
Decoder (symmetric): Latent (32) → Dense (128, ReLU)
→ Dense (512, ReLU) → Dense (346, linear).

Training setup:
The autoencoder was trained with **Mean Squared Error (MSE)** loss, a standard choice for reconstruction tasks as it directly measures squared differences between inputs and outputs. Optimization used the **Adam** optimizer (learning rate = 1e-4), which combines momentum and adaptive learning rates for efficient convergence. A **ReduceLROnPlateau** scheduler (factor = 0.5, patience = 10, min_lr = 1e-6) halves the learning rate if validation loss fails to improve for 10 epochs (helping the model escape shallow minima), but never drops it below 1e-6. Training was run for up to **500 epochs**, with **early stopping** based on validation loss to halt before overfitting. A **batch size of 512** was used, with a **90%/10% train/validation split** to monitor generalization. **Checkpointing** was applied so that only the model achieving the **lowest validation loss** was saved for downstream clustering experiments.

Latent representation: The resulting compressed space has shape **(10,299 × 32)**.
This design balances expressiveness with efficiency: the two-layer encoder–decoder learns hierarchical compression while the latent size (32) preserves structure without overfitting. The AE consistently converged with stable training and validation losses, and provided latent vectors Z that were used for downstream clustering.

Results:
After compressing the dataset to 32-D latent vectors, clustering experiments were repeated. The latent representation preserved most of the important variance while filtering redundancy, allowing cleaner separations:

HDBSCAN (direct on Z) → two clusters (sizes 5,624 and 4,675), with silhouette = 0.555, ARI = 0.318, NMI = 0.544.
UMAP → HDBSCAN (on Z) → stronger separation, silhouette = 0.848 (highest overall), ARI = 0.332, NMI = 0.555, again yielding two macro-clusters corresponding to static vs locomotion.
Spectral clustering (k=6 on Z) → highest ARI = 0.518 and NMI = 0.635 among single-stage methods, but with very low silhouette = 0.113, showing overlap despite label agreement.

Table 4. Clustering on 32-D AE Latent (Z).

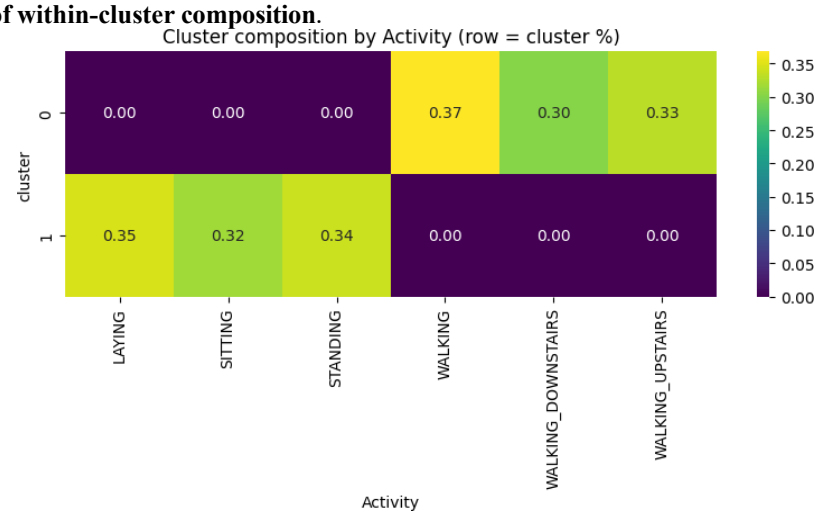
Algorithm	Silhouette	ARI	NMI
UMAP+HDBSCAN (nn=50, md=0.1, mcs=100, ms=10)	**0.848196**	0.3321	0.5549
HDBSCAN (latent direct)	0.554873	0.3184	0.5439
UMAP+HDBSCAN (nn=50, md=0.0, mcs=100, ms=10)	0.413808	0.3299	0.5448
Agglomerative (latent)	0.374343	0.3033	0.4717
KMeans (latent)	0.355214	0.2993	0.4659
MiniBatchKMeans (latent)	0.170653	0.3289	0.4743
GMM (latent)	0.136580	0.2670	0.4357
Spectral (latent)	0.113216	**0.5178**	**0.6354**

Two-Level Clustering: Broad Groups and Subclusters
Given the above findings, we adopted a **two-level clustering approach** to fully uncover the activity structure:

Level 1 – Broad Cluster Separation: Identify the major coarse clusters in the data. HDBSCAN (with AE latent space + UMAP preprocessing) naturally gave 2 clusters (and zero noise) as the optimal split. We designate these as Cluster 0 and Cluster 1. Upon examining their composition, Cluster 1 contained predominantly the static postures (*LAYING,*

SITTING, STANDING) while Cluster 1 contained the ambulatory activities (*WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS*). Indeed,
of the 5624 samples in Cluster 1, about 99% were from
{laying, sitting, standing} and only a negligible few from movement classes; conversely, Cluster 0's 4675 samples were ~99% from {walking, walking_upstairs, walking_downstairs}. This confirms that the first principal division of the data is **static vs. dynamic activities**. Each of these broad clusters still mixes multiple activities (purity
~0.35 for each cluster, since Cluster 1 is roughly one-third each laying/sitting/standing, and Cluster 0 one-third each of the walking categories).

Figure 1. **Heatmap of within-cluster composition**.



Level 2 – Subcluster Discovery within Each Group: We then analyzed each broad cluster separately to see if they further break down into the actual distinct activities. We applied dimensionality reduction and clustering *within* Cluster 0 and Cluster 1 independently.

Static cluster (Cluster 1): mainly laying, sitting, standing): Focusing only on the 5624 static samples, we ran UMAP (embedding just this subset) followed by clustering (k-means, GMM, spectral, etc.) to find subclusters. We found that a 3-cluster solution is appropriate (since there are three actual static activities). The best separation among static activities was achieved by UMAP + spectral clustering with k=3, which yielded an ARI ≈ 0.514 and NMI
≈ 0.623 when comparing to the true labels of the static samples. In this clustering, one subcluster was almost purely **LAYING**, and the other two subclusters contained mixes of **SITTING** and **STANDING**. In fact, one cluster was 98.8% LAYING (essentially isolating nearly all laying samples), while the remaining two clusters split the upright postures. One of those was roughly a balanced mix of sitting and standing (~47% sitting, 51% standing), and the other was a very small cluster (only 48 samples) that also consisted primarily of LAYING. This tiny cluster likely captured an outlier subset of laying posture data (perhaps instances with a different orientation or sensor placement). Essentially, the algorithm found that *laying (lying down) is distinctly different and easy to cluster on its own*, whereas sitting vs. standing are more similar to each other and sometimes got grouped together. Another method (t-SNE + GMM with 3 components on Cluster 0) similarly produced one cluster
~92% LAYING, and two clusters that split sitting/standing with about 54–57% purity for those classes, indicating those two activities are harder to separate without supervision.
Overall, unsupervised methods consistently distinguished **lying down** from **upright postures**, but tended to intermingle **sitting** and **standing** to some degree.

Dynamic cluster (Cluster 0: walking and stair activities): We performed a similar sub-clustering on Cluster 1 (which contains the three walking-related activities). Here the best separation was achieved by UMAP followed by k-means (k=3) on the dynamic subset. This produced three subclusters that corresponded roughly to: (a) mostly **Walking Upstairs**, (b) mostly **Walking Downstairs**, and
(c) a mix dominated by **Walking** (level ground). Specifically, one subcluster (call it Cluster 1.0) had ~81.5% of its samples as WALKING_UPSTAIRS (with a few walking and almost no downstairs instances) – clearly an "**Upstairs**" cluster. Another subcluster (Cluster 1.2) was
~50% WALKING_DOWNSTAIRS (with ~32% walking,
~17% upstairs), thus representing a "**Downstairs**"-dominant cluster. The third subcluster (Cluster 1.1) contained mainly **WALKING** on level ground (about 48% of that cluster), but also a substantial minority of upstairs and downstairs instances (~32% and ~19%, respectively), making it a more mixed cluster. This suggests that walking on flat terrain was not as cleanly separable – some windows of the stair activities were grouped with flat walking, possibly when the movement pattern was not distinctly stair-like (e.g., transitional moments or similar sensor signatures).

The subcluster purity for Cluster 1 was lower overall than for Cluster 0's case of isolating laying. For instance, the

upstairs-focused cluster was ~82% pure for WALKING_UPSTAIRS, the downstairs cluster ~51% pure for WALKING_DOWNSTAIRS, and the mixed cluster
~48% pure for WALKING. The ARI for the three clusters vs. true labels in Cluster 1 was only 0.11 (NMI ~0.14) for the k-means solution, reflecting that one cluster was mixed. (We note that a GMM on this dynamic cluster gave a higher ARI ~0.28, NMI ~0.30 – indicating it aligned with actual classes a bit better – but its clusters were less cleanly separated, silhouette 0.33 vs 0.39 for k-means. We chose the k-means result for interpretation because it yielded clearly interpretable clusters, e.g., one capturing almost all "Upstairs" instances, etc., even if one cluster was a blend of walking types.) This again underscores that purely unsupervised clustering can identify the major modes (up vs. down stairs), but may blur boundaries (flat walking vs. stairs) where activities share similarities.

In total, this two-level process revealed 5–6 meaningful clusters corresponding to the original activities:
A distinct cluster for **Laying** (found within the static group).A cluster for **Sitting/Standing combined** (the algorithm often grouped these; one could further attempt to split them with more focus, but unsupervised separation is difficult).A distinct small cluster of some **Laying** instances (an outlier subset of Laying posture).A cluster for **Walking Upstairs** (dominant in one dynamic subcluster).A cluster for **Walking Downstairs** (dominant in another subcluster).A cluster primarily for **Walking (level ground)**, though mixed with some stair instances (the third dynamic subcluster).

This hierarchical unsupervised approach essentially recovered the two fundamental categories (stationary vs moving) at the top level, and within each, it roughly recovered the finer categories (with some merging of the most similar pair, sitting/standing). Next, we delve into qualitative analysis to interpret these clusters: what features distinguish them, and how they correspond to the physical differences between activities.

Qualitative Analysis of Clusters and Feature Insights
We analyze two unsupervised clusters consistent with *static* and *dynamic* behavior: **Cluster 1 (STATIC, n=5,624)** and **Cluster 0 (DYNAMIC, n=4,675)**. To minimize redundancy, features were selected by low pairwise dependence (absolute Spearman $|\rho| \lesssim 0.25$ on the combined rows). Reported statistics are per-cluster **p25 / p50 / p75** ((25th, 50th, and 75th percentiles) and **SD** (Standard Deviation); separation is assessed by contrasts in **dispersion** (IQR (Inter-quartile Range), SD) and only secondarily by mean shifts.

1. Jerk means (primary drivers of separation)
Across all four "jerk mean" features (Means of the time-domain **jerk** signals—time derivatives of body acceleration (AccJerk) or gyroscope (GyroJerk)—along the **Y/Z** axes, i.e., the average rapid **linear** (AccJerk) and **rotational** (GyroJerk) changes), **cluster means are near zero**, yet **dispersion differs by an order of magnitude**. This yields a tight, near-origin STATIC core versus a broad DYNAMIC shell in feature space.

tBodyGyroJerk-mean(Y)
STATIC: −0.02 / **0.02** / 0.05, SD = 0.18 → IQR = 0.07
DYNAMIC: −0.86 / **−0.03** / 0.80, SD = 1.47 → IQR = 1.66
IQR ratio ≈ 24×, SD ratio ≈ 8×.
tBodyGyroJerk-mean(Z)
STATIC: −0.04 / **0.00** / 0.04, SD = 0.23 → IQR = 0.08
DYNAMIC: −0.85 / **0.03** / 0.87, SD = 1.46 → IQR = 1.72
IQR ratio ≈ 22×, SD ratio ≈ 6×.
tBodyAccJerk-mean(Y)
STATIC: −0.02 / **0.02** / 0.06, SD = 0.32 → IQR = 0.08
DYNAMIC: −0.88 / **−0.01** / 0.86, SD = 1.44 → IQR = 1.74
IQR ratio ≈ 22×, SD ratio ≈ 4.5×.
tBodyAccJerk-mean(Z)
STATIC: −0.04 / **0.02** / 0.08, SD = 0.38 → IQR = 0.12
DYNAMIC: −0.83 / **0.00** / 0.82, SD = 1.43 → IQR = 1.65
IQR ratio ≈ 14×, SD ratio ≈ 3.8×.
Dynamic windows include alternating phases that keep means near zero but produce **large moment-to-moment excursions**; STATIC windows exhibit **minimal jerk**. These axes therefore create a **radial dispersion contrast** that unsupervised methods can easily separate.

2. Body-gyro means ((secondary but consistent)
Means of the time-domain **gyroscope signals** (tBodyGyro) along the **Y** and **Z** axes, i.e., the average angular around those axes.
tBodyGyro-mean(Y)
STATIC: −0.08 / **−0.00** / 0.06, SD = 0.63 → IQR = 0.14
DYNAMIC: −0.61 / **0.00** / 0.67, SD = 1.32 → IQR = 1.28
IQR ratio ≈ 9×, SD ratio ≈ 2.1×.

tBodyGyro-mean(Z)
STATIC: −0.07 / **−0.01** / 0.05, SD = 0.85 → IQR = 0.12
DYNAMIC: −0.47 / **−0.03** / 0.35, SD = 1.15 → IQR = 0.82
IQR ratio ≈ 7×, SD ratio ≈ 1.35×.

Gyro means thicken the DYNAMIC cluster along additional, **low-correlated** axes, reinforcing the compact-vs-broad geometry.

3. Angles relative to gravity (range-of-motion contrast)
Angles between the **mean body acceleration/jerk or gyro jerk vectors** and the **gravity vector**, capturing body orientation and range-of-motion relative to gravity.
angle(tBodyAccMean, gravity)
STATIC: −0.17 / **−0.00** / 0.19, SD = 0.52 → IQR = 0.36
DYNAMIC: −1.06 / **0.02** / 1.01, SD = 1.37 → IQR = 2.07
IQR ratio ≈ 5.8×.
angle(tBodyAccJerkMean, gravityMean)
STATIC: −0.35 / **0.02** / 0.38, SD = 0.67 → IQR = 0.73
DYNAMIC: −1.22 / **−0.05** / 1.19, SD = 1.29 → IQR = 2.41
IQR ratio ≈ 3.3×.
angle(tBodyGyroJerkMean, gravityMean)
STATIC: −0.67 / **0.03** / 0.68, SD = 0.92 → IQR = 1.35
DYNAMIC: −0.94 / **−0.04** / 0.92, SD = 1.09 → IQR = 1.86
IQR ratio ≈ 1.4×.

DYNAMIC windows occupy a **broader orientation envelope** than STATIC, adding separability along angle dimensions that are only weakly correlated with jerk/gyro means.

4. Cross-axis correlations (weak for the static–dynamic split)
Measure the linear relationship between pairs of axes (X–Z or Y–Z) for gyroscope jerk or gravity signals, capturing how movements along one axis co-vary with another.
tBodyGyroJerk-corr(X,Z): medians 0.01 (STATIC) vs
−0.04 (DYNAMIC), SD ≈ 1.0; **small effect** (d ≈ 0.09).
tBodyGyroJerk-corr(Y,Z): medians −0.10 (STATIC) vs
+0.10 (DYNAMIC), SD ≈ 1.0; **small effect** (d ≈ −0.16).
tGravityAcc-corr(Y,Z) (Correlation between the gravity acceleration signals along the Y and Z axes, reflecting torso tilt and coupling of vertical–anteroposterior orientation relative to gravity) **:** medians 0.22 (STATIC) vs 0.06 (DYNAMIC); effect ≈ 0.

These features contribute little to the *binary* static–dynamic separation in this dataset; they have been more informative within posture or stair-direction splits in your other tables.

INTRACLUSTER INSIGHTS
Static Parent → Subcluster 1.0 (mixed SITTING/STANDING)
Two cross-axis correlations organize this subcluster:
tGravityAcc-corr(X,Z) shows a clear sign bifurcation across posture:
SITTING: mean −0.33, median −0.86, IQR ≈ [−1.20, +0.60]
STANDING: mean +0.32, median +0.61, IQR ≈ [−0.86, +1.44]
This produces a bimodal spread around zero inside 1.0: strongly negative values cluster with SITTING; strongly positive values with STANDING. The wide IQRs confirm overlap, but the opposite-sign medians give 1.0 its posture-shaped geometry. tBodyGyroJerk-corr(X,Z) provides the supporting axis:

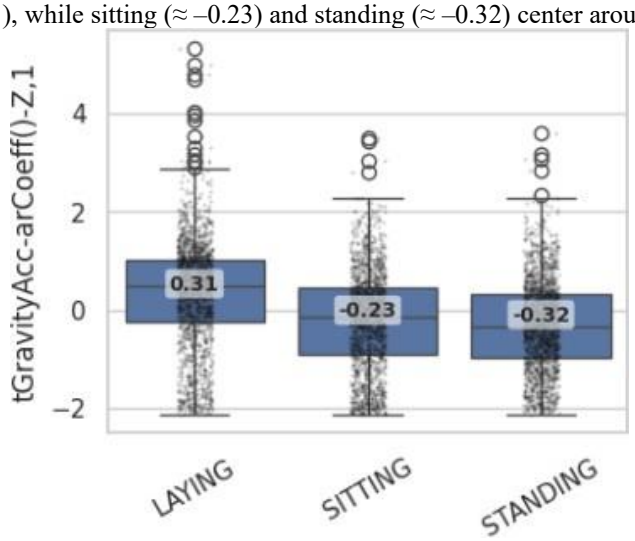
SITTING: mean +0.11, median +0.10, IQR ≈ [−0.50, +0.69]
STANDING: mean −0.21, median −0.28, IQR ≈ [−0.85, +0.38]
Medians differ in sign while spreads overlap. Inside 1.0, this feature behaves like a secondary spine that tilts the mixed region toward one posture or the other when gravity corr(X,Z) is near its overlap band.
Other features are comparatively neutral for shaping 1.0's sit/stand structure:
tGravityAcc-corr(Y,Z) (means 0.34 vs 0.41, medians 0.77 vs 0.86) is similarly positive for both.
angle(tBodyAccMean,gravity) (medians +0.01 vs −0.01) and angle(tBodyGyroJerkMean,gravityMean) (medians −0.04 vs +0.06) show small center shifts with broad spread.

tBodyAccJerk-mean(X/Z) and tBodyGyro-mean(Z) sit near zero in both groups.

tGravityAcc-corr(X,Z) (Pearson correlation between the time-domain gravity acceleration components on X and Z (low-pass accelerometer), capturing cross-axis orientation coupling.) reflects torso orientation relative to gravity.

Seated postures often involve posterior tilt or altered alignment, pulling X and Z gravity components into opposing variation (negative correlation). Upright stance aligns segments differently, producing a positive coupling.
tBodyGyroJerk-corr(X,Z) reflects tiny corrective jerks: counter-axis corrections are more common in stance (negative), while in-phase micro-jerks are more typical when seated (positive). The sign patterns you observe match these mechanisms.

Figure 2. Distribution of *tGravityAcc-arCoeff(Z,1)* across static postures.As shown, laying windows cluster around positive values (median ≈ 0.31), while sitting (≈ –0.23) and standing (≈ –0.32) center around negative values.



Locomotion parent → Subcluster 0.1 (balanced WALK/UP/DOWN)
0.1 carries the continuous spectrum of level and stair walking:
tGravityAcc-corr(Y,Z) spans three tendencies within the subcluster
WALK: median ~−0.02 (near zero) UP: median ~−0.16 (negative tail) DOWN: median ~+0.41 (positive tail)

With SDs ≈ 0.80–0.85, 0.1 looks like a tri-skew corridor: a central mass near 0 (level), a negative lobe (upstairs), and a positive lobe (downstairs). tBodyAccJerkMag-arCoeff(3) follows a monotone shape trend (means/medians shift from more negative in WALK toward less negative in UP and then DOWN), indicating periodicity shape changes on stairs. fBodyGyro-bandsEnergy(57–64) is higher on stairs on average (Walk → Up → Down), consistent with additional high-frequency components during stair negotiation. Medians remain <0 but the mean trend aligns with stair intensity. The Y–Z gravity coupling captures the pitch bias: uphill tends to back-tilt, downhill forward-tilts, while level walking stays close to neutral. Stair negotiation also introduces sharper, more frequent angular changes,
raising high-band gyro energy and subtly altering the autocorrelation shape of jerk magnitude.

Locomotion parent → Subcluster 0.0 (UP-heavy; WALK vs UP slice)
Within 0.0, uphill patterns visibly shift several distributions: fBodyGyro-min(X) is consistently higher for UP (median +0.27) than WALK (~0.00), with the upper tail much larger in UP (p75 1.08 vs 0.48). This implies fewer large negative gyro excursions along X during ascent.
tBodyGyroJerk-mean(Y) flips sign (UP median +0.08 vs WALK −0.18), indicating more in-phase Y-jerk in the uphill steps. tBodyGyro-mean(Z) is more negative for UP (~−0.17) than WALK (~0.00), suggesting a persistent rotational bias during climbing. tBodyAccJerk-mean(Z) is slightly more positive for WALK (median +0.05) than UP (0.00); fBodyAcc bands(57–64) are modestly higher for UP (medians less negative), consistent with increased
high-frequency content. Ascending requires controlled elevation and reduced extreme counter-swings, lifting the minimum X-gyro. The positive Y-jerk mean suggests a more coordinated upward phase in each cycle. The more negative Z-gyro mean aligns with a sustained rotational posture bias during ascent.

Locomotion parent → Subcluster 0.2 (DOWN-heavy; WALK vs DOWN slice)
Within 0.2, downhill patterns pull features to the opposite sides:
tGravityAcc-corr(Y,Z) moves more positive (DOWN median +0.40 vs WALK +0.18), capturing a forward-lean tendency. tBodyAcc-mean(X) and tBodyAcc-mean(Y) are more positive for DOWN (medians 0.17 and ~+0.05) than WALK (0.08 and ~0.00), indicating a net forward component during descent. tBodyAccJerk-mean(Z) flips sign (DOWN +0.03 vs WALK −0.03), while tBodyGyroJerk-mean(Y) is more negative for DOWN (−0.15 vs −0.02). Together these suggest sharper deceleration/impact modulation on descent. fBodyAccJerk-min(Y) tends to be lower for DOWN (median 0.63) than WALK (0.77), consistent with deeper negative jerk troughs in Y during downhill steps.
Descending emphasizes controlled forward pitch and impact absorption, which pushes gravity coupling positive, increases forward components in body acceleration, and yields more negative Y-gyro jerk alongside deeper jerk

minima.
In the standardized space, STATIC forms a compact, near-origin core across multiple low-correlated axes;
DYNAMIC expands into a multi-axial, high-variance cloud. Unsupervised methods naturally exploit this radial dispersion contrast—no large mean offset is required. The selected features are **mutually low-correlated** and exhibit **large dispersion increases** for DYNAMIC across several independent axes (jerk means, gyro means, angles).
Consequently, in the standardized feature space, STATIC forms a **compact cluster near the origin**, whereas DYNAMIC expands into a **high-variance, multi-axial cloud**. This geometry is precisely what distance- or graph-based clustering exploits to separate the two groups
without relying on mean shifts.

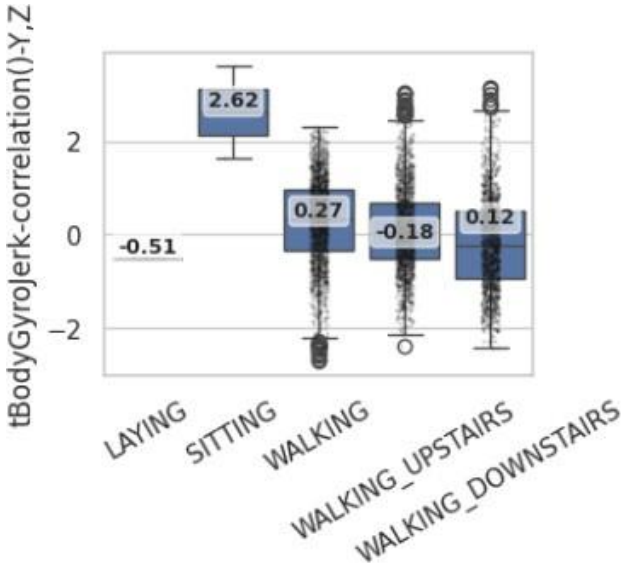


Figure 3. Distribution of *tBodyGyroJerk-correlation(Y,Z)* across activities.

Among dynamic classes, this feature separates level walking (≈ 0.27) from upstairs (≈ –0.18) and downstairs (≈ –0.12), indicating distinct gyro-jerk correlation patterns for different walking modes.

Discussion
The discussion section interprets the clustering results beyond raw metrics, explaining what the discovered patterns mean in the context of human activities. It highlights the hierarchical nature of activity groupings, where broad static–dynamic separation underlies finer posture and movement distinctions. This analysis is essential to connect algorithmic outcomes with real-world understanding of sensor signals and activity taxonomy.

Dominant Latent Structure – Static vs. Dynamic: The most prominent inherent pattern in this HAR dataset is the separation between non-moving (static) postures and moving activities. Unsupervised algorithms consistently found this division: methods that automatically determine cluster count (like HDBSCAN) or optimize cluster separation tended to produce two clusters corresponding to this split. This was evidenced by the high silhouette scores for two-cluster solutions and by the fact that those clusters respectively contained >99% static or dynamic instances. This insight aligns with our understanding that sensor data from a person sitting/standing still versus walking are fundamentally different in magnitude and variation.

Hierarchical Nature of Activity Clusters: The activities form a natural hierarchy: at the top level are broad categories (stationary vs. moving), and at the next level are specific activity types. Purely "flat" clustering into six groups was less successful unless guided by the knowledge of needing six clusters. However, by recognizing the two-level structure, we effectively recovered the six classes: the static cluster split into three subclusters (with some blurring between sitting and standing), and the dynamic cluster split into three subclusters (distinguishing upstairs, downstairs, and level walking, with some mixing of the latter two). This demonstrates that a hierarchical clustering approach can mirror the taxonomy of activities: first discriminate gross movement vs. no movement, then finer distinctions of how one is moving or what posture one holds.

Role of Dimensionality Reduction: Non-linear embedding techniques like UMAP greatly facilitated finding these structures. In the original high-dimensional space, the clusters were present but harder for algorithms to isolate (resulting in lower performance). UMAP created a representation where distances between points more closely reflected the true activity differences. This not only improved clustering metrics but also allowed visualization of the cluster structure. PCA, by contrast, was less effective in separating activities, likely because the variance captured by PCA is not solely due to activity differences (sensor noise or subject-specific differences might consume top components). UMAP (with an appropriate neighborhood parameter) seemed to capture the manifold of human motions better. The autoencoder, which learned an even more compact representation, reinforced the utility of

non-linear feature learning: while it didn't drastically change the cluster outcome, it confirmed that the two-cluster separation is robust and that more subtle class distinctions still require either hierarchical handling or additional context to unravel.

Utility of Exploratory Unsupervised Analysis: The process highlighted the value of Exploratory Data Analysis (EDA) tools for unsupervised learning:

Box plots of feature values for each (sub)cluster were instrumental in diagnosing why clusters formed and in assigning meaning to them. By examining distributions of candidate features, we translated the cluster differences into human-understandable terms (orientation stability, movement frequency, etc.). Having removed highly correlated features upfront made this interpretation easier, since each examined feature provided unique information. Otherwise, redundant features could have obscured the patterns. The combination of domain knowledge (knowing what each sensor feature measures physically) with data-driven cluster assignments made it possible to draw the qualitative conclusions above. This is a critical step in unsupervised learning – clusters must be interpreted to be useful. Our analysis serves as an example of how to bridge the gap between raw clustering output and insightful understanding of the data.

Clustering Algorithm Comparisons: Among clustering algorithms:

Density-based methods (HDBSCAN) excelled at finding the obvious structure (static vs dynamic) without being forced to split further. This yielded very clean, well-separated clusters (high silhouette), though coarse in granularity.

Centroid-based methods (K-Means, Mini-Batch KMeans) needed the number of clusters set; when k=6 they moderately matched the six activities but had trouble due to overlapping classes (e.g., sitting vs standing are too similar for clear centroid separation). K-Means performed better after UMAP embedding, showing that with a good representation it can capture classes reasonably (in our case ARI ~0.54 on 2D UMAP).

Gaussian Mixtures (GMM) similarly benefited from the embedding; interestingly GMM achieved slightly higher ARI (~0.55) than K-Means on the UMAP data, suggesting some activities might be better represented as overlapping Gaussian blobs rather than hard Voronoi partitions.

Spectral clustering was the best at aligning clusters to true labels in the original space (highest ARI ~0.48 for k=6 in raw features, ~0.52 in latent space), implying the activity differentiation is more evident from a graph affinity perspective. However, its clusters were not as separated in feature space (low silhouette), and spectral clustering is computationally heavier for larger data.

Agglomerative clustering (Ward's linkage) gave decent but not top results: slightly above k-means in raw space ARI, but below GMM/spectral in embedded space. Its advantage is simplicity and deterministic nature, but it can be sensitive to noise and the assumption of hierarchical structure.

DBSCAN (the non-hierarchical version) was not very useful on this dataset without supervised parameter tuning; in high-D space it either merged everything into one cluster or created many tiny clusters. In lower-D embeddings, it still did not cleanly find the six groups (often merging some and labeling others as noise). It's better suited for cases where clusters have irregular shape or there is true noise – here every point belongs to some activity class, so labeling points as "noise" is not conceptually appropriate.

Ability to Identify Rare or Outlier Clusters: We noticed the clustering occasionally isolated a very small cluster (e.g., 48 laying samples separated from the main laying cluster, or some walking segments that formed their own tiny cluster depending on the method). These might correspond to outlier behavior or transition periods not well represented in the majority of data. For instance, the tiny laying cluster could indicate a subset of "laying" windows where the person's orientation was slightly different (perhaps lying on their side vs. back) leading to different sensor patterns. The unsupervised approach can flag such anomalies for further inspection – a benefit that supervised models might gloss over by focusing on majority patterns. In a practical sense, this means unsupervised clustering could help discover *unknown modes or unusual activities* present in the data without explicit labels.

Limitations

Despite generally corresponding clusters, the unsupervised approach did not perfectly separate all activities:

Sitting vs Standing: These remained partially conflated in our clusters. They are inherently similar in sensor readings (both involve an upright stationary body). The differences (postural sway, slight movements) are subtle. Without explicit label information or additional constraints, a clustering algorithm is likely to group many sitting and standing instances together, as we observed. Additional features (perhaps from other sensors for posture) or multi-step clustering focusing specifically on separating those clusters would be needed to fully split them. In practical terms, an unsupervised model might consider them one cluster of "upright stationary" behavior.

Walking vs Variations (speed/transitions): Our data did not have jogging or different speeds of walking; if it did, we might see speed-based clustering. Even within walking, our clusters hinted that faster vs slower walking, or walking vs transitional moments (starting/stopping), could cause mixing. For example, the mixed cluster 1.1 had some upstairs/downstairs instances which could be misclustered transitional windows (not strongly stair-like). So activities that are part of a continuum (speed or intensity) can blur together unsupervised.

Mixing of Stair and Level Walking: The cluster that was primarily level walking still contained a notable fraction of upstairs/downstairs data. These could be windows where a person is in the process of starting or ending a stair climb (thus not strongly characterized as stairs in that short window), or simply an artifact of clustering not having enough distinguishing power in those borderline cases. A more complex model or a larger feature set (e.g., including time continuity constraints or additional sensors) might separate these more cleanly.

No use of time/sequence information: We treated each data window independently. In reality, activities form sequences (one typically goes from walking to climbing stairs to walking, etc.). Incorporating temporal continuity (with Hidden Markov Models or Conditional Random Fields or a sliding window clustering that accounts for sequence likelihood) could improve separation of ambiguous windows by using context. However, our aim here was purely static clustering of windows to see what structure emerges without sequence modeling.

Generality: These insights are specific to the devices and population of this dataset (smartphone on waist, 30 subjects). Different sensor setups or populations might yield different cluster structures (e.g., if some activities were more variable or if sensor placement changed, the unsupervised grouping might differ). Nonetheless, the static vs dynamic split is likely universal for motion data, and certain feature patterns (like those we identified) would probably appear in any inertial sensor-based HAR data.

IV. CONCLUSION

We conducted an in-depth unsupervised analysis of a human activity recognition dataset, uncovering its inherent structure *without using activity labels* in the clustering process. Our findings show that the dominant separation in the data is between **static postures** and **dynamic movements**. By applying a hierarchical clustering strategy – first splitting the data into broad clusters, then sub-clustering each – we were able to approximately recover the six known activity classes. Non-linear dimensionality reduction (especially UMAP) greatly aided the clustering, making the natural groupings more separable. Among clustering algorithms, HDBSCAN was effective for finding the top-level split, while spectral clustering and k-means (on embedded data) helped isolate finer groupings..

In summary, unsupervised learning was able to discover the major patterns in the HAR data, grouping similar activities together and highlighting subtle differences. This approach could be useful for automated pattern discovery in activity data – for example, to identify unknown modes of activity or to pre-process data before applying supervised learning. The main limitation is that very similar activities (or subtle variations of one activity) can be hard to separate without guidance. Future work could explore incorporating sequence information or constraints to improve discrimination between such activities. Nevertheless, this experiment demonstrates a successful exploratory analysis: we gained insights into inherent activity patterns and the features underlying them, using clustering and dimensionality reduction as our microscope into the data.

ACKNOWLEDGMENT

The authors wish to acknowledge the use of *ChatGPT* in the preparation of this paper. This tool was used to assist with summarizing experimental results and improving the phrasing of technical content. The paper remains an accurate representation of the authors' underlying work and novel intellectual contributions. The authors further clarify that this work has been carried out independently of VIT, and the institution bears no responsibility for the research presented here.

REFERENCES

[1] Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A*, 374(2065), 20150202.

[2] Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2579-2605.

[3] McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

[4] Anguita, D., Ghio, A., Oneto, L., Parra, X., & Reyes-Ortiz, J. L. (2013). A public domain dataset for human activity recognition using smartphones. *21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*,437-442.

[5] Amrani, M., Djenouri, Y., Habbas, Z., & Belhadi, A. (2022). Deep inertial sensory clustering for human activity recognition. *IEEE Sensors Journal*, 22(13), 12688-12697.

[6] Ordóñez, F. J., & Roggen, D. (2016). Deep convolutional and LSTM recurrent neural networks for multimodal earable activity recognition. *Sensors*, 16(1),115.
[7] Kwapisz, J. R., Weiss, G. M., & Moore, S. A. (2011). Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2), 74-82.
[8] Xu, H., Liu, J., Tan, H., & Zhang, Y. (2025). DCAM-Net: A deep convolution attention MLP network for smartphone-based human activity recognition. *Expert Systems with Applications*, 238, 121907.
[9] Sculley, D. (2010). Web-scale k-means clustering. *Proceedings of the 19th International Conference on World Wide Web (WWW)*, 1177–1178.
[10] McLachlan, G., & Peel, D. (2000). *Finite Mixture Models*. Wiley.
[11] Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American StatisticalAssociation*, 58(301), 236–244.
[12] Campello, R. J. G. B., Moulavi, D., & Sander, J. (2013). Density-based clustering based on hierarchical density estimates. *Advances in Knowledge Discovery and Data Mining (PAKDD)*, 160–172. (See also the extended journal version: Campello et al., 2015, *ACM Transactions on Knowledge Discovery from Data*, 10(1), 5.)
[13] Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems (NeurIPS)*, 14, 849–856.
[14] Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504–507.