
MoPFormer: Motion-Primitive Transformer for Wearable-Sensor Activity Recognition

Anonymous Author(s)

Affiliation

Address

email

Abstract

Human Activity Recognition (HAR) with wearable sensors is challenged by limited interpretability, which significantly impacts cross-dataset generalization. To address this challenge, we propose Motion-Primitive Transformer (MoPFormer), a novel self-supervised framework that enhances interpretability by tokenizing inertial measurement unit signals into semantically meaningful motion primitives and leverages a Transformer architecture to learn rich temporal representations. MoPFormer comprises two-stages. first stage is to partition multi-channel sensor streams into short segments and quantizing them into discrete “motion primitive” codewords, while the second stage enriches those tokenized sequences through a context-aware embedding module and then processes them with a Transformer encoder. The proposed MoPFormer can be pre-trained using a masked motion-modeling objective that reconstructs missing primitives, enabling it to develop robust representations across diverse sensor configurations. Experiments on six HAR benchmarks demonstrate that MoPFormer not only outperforms state-of-the-art methods but also successfully generalizes across multiple datasets. Most importantly, the learned motion primitives significantly enhance both interpretability and cross-dataset performance by capturing fundamental movement patterns that remain consistent across similar activities regardless of dataset origin.

1 Introduction

Human Activity Recognition (HAR) has a wide range of applications and can be achieved through various methods [42, 4]. While vision-based and environmental sensor-based methods have been extensively explored [23, 18], systems leveraging Inertial Measurement Units (IMUs) offer distinct advantages, including compact size, low power consumption, and minimal computational overhead for data acquisition and preprocessing. IMUs also inherently preserve privacy compared to visual alternatives [10, 31] and are robust to environmental factors like lighting variations and occlusions [25]. Such characteristics make IMUs a highly suitable sensing modality for capturing human motion across diverse real-world scenarios.

Despite these strengths, IMU-based HAR confronts a significant hurdle: interpretability. IMU data, comprising multi-channel time series of linear acceleration and angular velocity, is inherently less intuitive for humans to interpret than video data, making it hard to understand what the model has learned [19]. Moreover, many human activities exhibit hierarchical structures, composed of more fundamental activities or movements. However, there is a common absence of fine-grained labels for the sub-activities that constitute these complex, hierarchical activities. For example, an activity like “washing hands” can be deconstructed into sub-activities such as “turning on the tap”, “applying soap”, scrubbing hands”, and “rinsing hands”. A HAR model might learn to identify “washing hands” by detecting the “turning on the tap” sub-activity, especially if it is a prominent

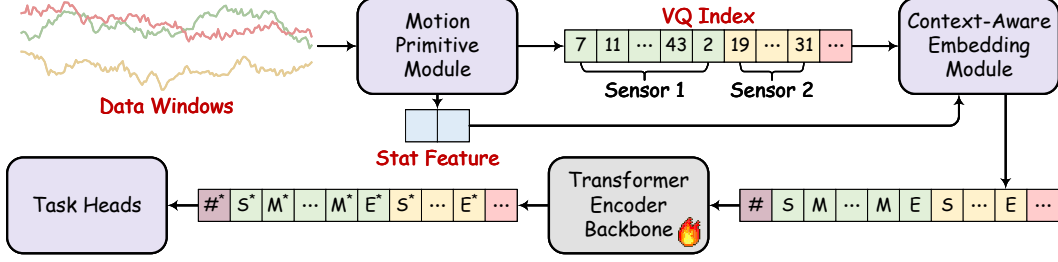


Figure 1: Architecture of the proposed MoPFormer model, showcasing the flow from raw data windows through tokenization, embedding, Transformer encoding, to task-specific heads.

signal. However, if the model encounters a different activity, such as “filling a water bottle”, which also begins with “turning on the tap”, it may struggle for HAR models to differentiate between the two activities. This is because the model might rely on the common initial sub-activity rather than grasping the complete, semantically meaningful sequence (e.g., the subsequent application of soap and scrubbing unique to handwashing) that defines the core of the activity. Each of these sub-activities might, in turn, be composed of even finer-grained movements. The black-box nature of contemporary HAR models often obscures whether these models are learning the underlying activity structure or merely exploiting superficial data correlations, thereby exacerbating the interpretability challenge [27, 3].

Another challenge in sensor-based HAR is data heterogeneity. Variations in sensor types, sampling rates, on-body placement (e.g., sensors worn on the wrist or the waist), subject characteristics (e.g., age, height, fitness level of the individual), and environmental conditions can introduce significant distributional shifts [32]. This heterogeneity impedes model generalization across different contexts. Consequently, developing strategies to address data heterogeneity is crucial for building robust HAR systems.

To address these challenges, we introduce **Motion-Primitive TransFormer (MoPFormer)**, a novel pre-training architecture for IMU-based HAR that significantly enhances model interpretability. Inspired by advancements in language modeling [20], MoPFormer conceptualizes IMU motion sequences as a series of “Motion Primitives” that serve as semantically meaningful, discrete building blocks of human activity. As illustrated in Fig. 1, our approach first divides raw IMU sequences into short segments, analogous to “words” in natural language, creating an interpretable vocabulary of fundamental movements. Each segment integrates data from various sensor channels at concurrent time points, forming feature vectors that are then arranged sequentially into a two-dimensional feature matrix. This construction not only mirrors the arrangement of words in a sentence but also enables transparent analysis of which primitive movements contribute to activity recognition decisions. To further enrich semantic understanding, we embed metadata for each sensor using a context-aware embedding module. The resulting representation allows for direct examination of motion primitive similarities, frequencies, and their distribution across different activities, providing unprecedented insights into model behavior. This comprehensive representation is subsequently processed by a Transformer backbone [48], enabling MoPFormer to effectively handle heterogeneous channel configurations and learn robust, interpretable features for diverse downstream tasks.

In summary, the primary contributions of this work are:

- We propose MoPFormer, an effective pre-training framework that achieves state-of-the-art performance in our experiments.
- We introduce a representation method that enables unified training across heterogeneous datasets.
- We extract motion primitives and provide a detailed interpretability analysis of these primitives.

2 Related Work

Traditional human activity recognition approaches train model parameters on different datasets separately, including statistical feature extraction methods and deep learning approaches. Deep learning methods include CNN-based architectures such as CALANet [33], COA-HAR [50], MA-CNN [37],

77 and SenseHAR [24], RNN-based approaches like DeepConvLSTM [30], and attention-based mod-
 78 els such as THAT [26] and PA-HAR [51]. Recent research has proposed general-purpose time series
 79 models applicable to various tasks, including classification, such as TimesNet [49], TSLANet [17],
 80 and FITS [52]. However, these methods are typically trained and tested on splits from the original
 81 datasets and are limited in their ability to achieve cross-dataset generalization.

82 Self-supervised human activity recognition methods perform representation learning through vari-
 83 ous approaches. Reconstruction-based methods such as TST [55] and TARNet [13] focus on rebuild-
 84 ing input signals. Contrastive learning methods such as TS2Vec [53], CL-HAR [35], DDLearn [36],
 85 TS-TCC [16], FOCAL [28], and ModCL [22] execute discriminative representation learning tasks.
 86 Other self-supervised objectives like BioBankSSL [55, 14] and Step2Heart [45] also contribute
 87 to learning time series representations. These methods typically learn feature representations on
 88 specific datasets and fine-tune classifier heads on the target dataset, but they can only perform on
 89 datasets with similar structures and still require labeled data from the target dataset.

90 Research on sensor-based HAR for composite activity recognition is relatively limited [11]. For in-
 91 stance, Chen et al. decomposed composite activities into multiple simple activities [11], where each
 92 simple activity is represented by a sequence of sensor signal segments. These segments are first
 93 fed into a CNN to extract representations for identifying simple activities. Concurrently, the CNN-
 94 extracted features from all segments are passed to an LSTM network to achieve high-level semantic
 95 activity classification. Similarly, prior work [12] inferred composite activities using estimated activ-
 96 ity sequences, where temporal correlations of simple activities were extracted for composite activity
 97 classification. Conversely, predicted composite activities were used to aid the derivation of simple
 98 activity sequences for the next time step, with predictions for both simple and composite activity
 99 sequences mutually updating during inference. While these studies present viable approaches for
 100 composite activity research, datasets with comprehensive, multi-level hierarchical labels for com-
 101 posite activities are scarce, and existing hierarchical annotations are often incomplete.

102 The concept of motion primitives has gained traction in robotics research [41, 9]. These primitives
 103 are fundamental movement patterns that can be sequenced and combined to generate or understand
 104 a diverse repertoire of complex human activities. For instance, Calvo et al. utilized eight such
 105 primitive movements, including “static” and partial “squat”, to train Hidden Markov Models for
 106 activity classification. Similarly, Sanzari et al. developed a framework for automatically discovering
 107 human motion primitives from 3D pose data by optimizing “motion flux” and then clustering the
 108 results [41]. However, the application of motion primitives to enhance the interpretability of IMU-
 109 based HAR has seen limited progress. This limitation stems largely from the inherently non-intuitive
 110 nature of IMU sensor data, which makes it challenging to define and interpret primitives directly
 111 from these signals.

112 Much of the research on interpretability in HAR has focused on attention mechanisms. The principle
 113 behind deep attention models is to weight input components, with the assumption that components
 114 assigned higher weights are more relevant to the recognition task and exert greater influence on the
 115 model’s decisions [43]. For example, Shen et al. [44] designed a segment-level attention method to
 116 determine which time periods contain more information; combined with a gated CNN, this segment-
 117 level attention can extract temporal dependencies. Zeng et al. [54] developed attention mechanisms
 118 from two perspectives: they first proposed sensor attention on the input to extract salient sensory
 119 patterns, and then applied temporal attention to LSTMs to filter out inactive data segments. Ma
 120 et al. [29] employed spatial and temporal attention mechanisms, extracting spatial dependencies
 121 by fusing patterns with self-attention. However, their interpretability often remains at a superficial
 122 level, indicating what the model focuses on, rather than why or how these features contribute to the
 123 recognition of complex, hierarchically structured activities.

124 3 Methodology

125 The architecture of MoPFormer, as illustrated in Fig. 2, comprises four modules: the motion prim-
 126 itive module, the context-aware embedding module, and two task heads. The motion primitive
 127 module first transforms multi-dimensional time-series data windows into index sequences from a
 128 fixed vocabulary. To further extract patterns, the context-aware embedding module extracts seman-
 129 tic information from contextual relationships and embeds high-dimensional vector sequences for the
 130 Transformer backbone. The MAE head, as a pre-training task, strengthens the context-aware embed-

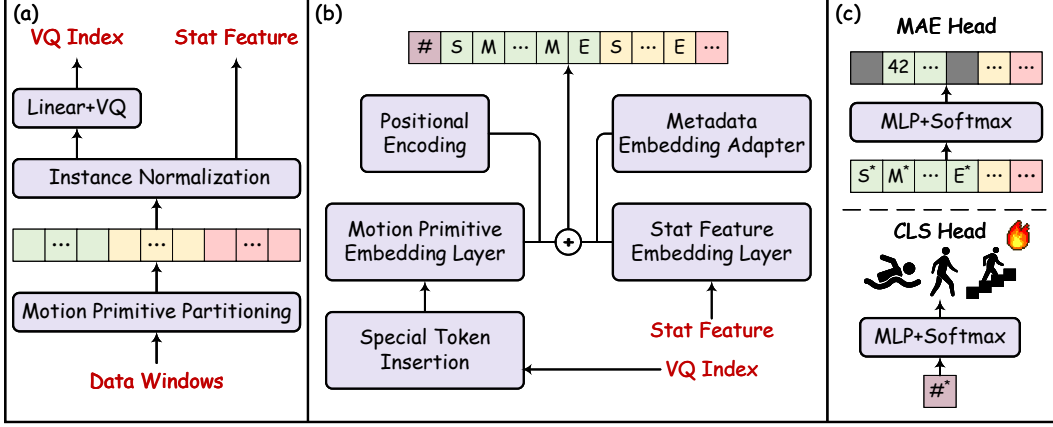


Figure 2: Detailed illustration of key modules in our motion-centric framework. (a) Motion Primitive Module: raw data windows from multiple sensor channels are partitioned into motion primitives and processed through instance normalization to generate VQ indices and statistical features. (b) Context-Aware Embedding Module: special tokens are inserted alongside masked motion embedding tokens (M) to form the complete input representation, which combines motion primitive embeddings, positional encodings, and statistical feature embeddings. (c) Task Heads: The diagram shows transformed features X^* representing the corresponding position vectors after Transformer Encoder processing. The MAE head utilizes M^* positions for masked token prediction during pretraining, while the trainable CLS head operates exclusively on the transformed [CLS] token representation for downstream task fine-tuning.

ding module’s ability to understand motion primitives, while the CLS head enables the Transformer backbone to learn effective features for classification. Fig. 2 illustrates the detailed architecture of each key module in our framework.

3.1 Motion Primitive Module

The Motion Primitive Module transforms raw data windows into discrete motion primitives, serving as the foundational, interpretable units for our motion-centric representation. For each data window of shape $T \times C$, where T is the number of time steps and C is the number of channels (e.g., accelerometer, gyroscope), we first partition the sequence into non-overlapping segments across each channel independently. Each channel is divided into segments of length L , resulting in $S = \lfloor T/L \rfloor$ segments per channel and a total of $S \times C$ segments per window.

Each segment undergoes a feature extraction process where we compute both low-level temporal patterns and statistical characteristics. For each segment $s_i \in \mathbb{R}^L$ (a single-channel segment), we apply instance normalization in Eq. (1) to standardize the input:

$$\hat{s}_i = \frac{s_i - \mu(s_i)}{\sigma(s_i) + \epsilon}, \quad (1)$$

where $\mu(s_i)$ and $\sigma(s_i)$ represent the mean and standard deviation of segment s_i , and ϵ is a small constant for numerical stability. This normalization removes sensor-specific scale differences, ensuring all segments are on a comparable scale before quantization.

The normalized segments are then encoded into discrete motion primitives using a Vector Quantization (VQ) [47] approach. We maintain a learnable codebook $\mathcal{Z} = \{z_1, z_2, \dots, z_K\}$, which contains K prototype vectors (or “motion primitives”), each of dimension L . As illustrated in Fig. 2(a), for each normalized segment \hat{s}_i , we compute its VQ index through Eq. (2) by finding the nearest prototype in the codebook:

$$q_i = \arg \min_{k \in \{1, 2, \dots, K\}} \|\hat{s}_i - z_k\|_2^2. \quad (2)$$

This quantization process maps continuous sensor data to discrete motion primitives, establishing a “vocabulary” of motion primitives. Alongside the VQ indices, we extract statistical features from

each segment (mean, variance) to capture complementary information that might be lost during quantization. These statistical features f_i are concatenated with the VQ indices to form the complete motion primitive representation.

The codebook is learned end-to-end through a commitment loss defined in Eq. (3) that encourages consistency between the input segments and their quantized representations:

$$\mathcal{L}_{VQ} = \|sg[\hat{s}_i] - z_{q_i}\|_2^2 + \beta \|\hat{s}_i - sg[z_{q_i}]\|_2^2, \quad (3)$$

where $sg[\cdot]$ denotes the stop-gradient operator and β is a hyperparameter controlling the commitment cost.

3.2 Context-Aware Embedding Module

The Context-Aware Embedding Module transforms the discrete motion primitive indices and their associated statistical features into rich, contextualized representations suitable for the Transformer backbone. This module aims to capture not only the type of motion primitive but also its intensity, variability, sensor origin, and temporal position. For each data window, the module processes the sequence of VQ indices $\{q_1, q_2, \dots, q_S\}$ and corresponding statistical features $\{f_1, f_2, \dots, f_S\}$ from all channels.

For the special tokens [MASK], [START], and [END], we assign reserved indices ($K + 1$, $K + 2$, and $K + 3$, respectively) in our vocabulary, ensuring they are consistently represented across different sensor channels. Using a learnable embedding matrix $E_{VQ} \in \mathbb{R}^{(K+3) \times D}$, where D is the embedding dimension, we embed the VQ indices through Eq. (4):

$$e_i^{VQ} = E_{VQ}[q_i]. \quad (4)$$

The [CLS] token is a separate learnable parameter outside the VQ embedding matrix.

The statistical features obtained during instance normalization (mean, variance) are projected into the same embedding space via a linear projection in Eq. (5):

$$e_i^{stat} = W_{stat}f_i + b_{stat}. \quad (5)$$

To incorporate sensor metadata (e.g., sensor type, mounting position) and enhance the model’s ability to generalize across different sensor configurations and distinguish between data from different sources, we introduce the Metadata Embedding Adapter for sensor-specific embeddings. For each segment i , its corresponding sensor channel c has associated metadata. This metadata is first converted into a fixed-length vector $e_c^{meta} \in \mathbb{R}^N$ using a pre-trained text embedding model. The Metadata Embedding Adapter uses a linear layer ($W_{adapter} \in \mathbb{R}^{D_{model} \times N}$, $b_{adapter} \in \mathbb{R}^{D_{model}}$), then maps this N -dimensional vector to our model’s embedding dimension D_{model} .

For each segment i , its complete token embedding e_i is formed by summing these constituent embeddings, as shown in Eq. (6). This fusion represents the motion primitive in a way that includes its quantized shape (e_i^{VQ}), magnitude/variability information (e_i^{stat}), and sensor context ($e_c^{adapter_emb}$):

$$e_i = e_i^{VQ} + e_i^{stat} + W_{adapter}(e_c^{meta}) + b_{adapter}, \quad (6)$$

where e_c^{meta} is the pretrained text embedding of dimension N for the channel c that segment i belongs to.

The model incorporates various special tokens to provide structural cues in the input sequence.

The final input sequence to the transformer backbone is structured as in Eq. (7):

$$X = [[CLS], [START], e_1^1, \dots, e_{n_1}^1, [END], \dots, [START], e_1^C, \dots, e_{n_C}^C, [END]], \quad (7)$$

where e_j^c represents the j -th embedding from sensor channel c , and n_c is the number of segments for channel c .

To provide temporal context, positional encodings are added through Eq. (8):

$$\hat{X} = X + P, \quad (8)$$

where $P \in \mathbb{R}^{S \times D}$ contains learnable position embeddings. Importantly, our positional encoding scheme assigns identical positional embeddings to data from different channels at the same time step, meaning that e_1^1 and e_1^C receive the same positional encoding if they represent data from the same time point but different channels.

196 3.3 Task Heads

197 As shown in Fig. 2(c), MoPFormer employs a dual-task learning approach with two specialized
198 heads built on top of the transformer encoder:

199 **Masked Auto-Encoding (MAE) Head.** During pre-training, we randomly mask a portion of the
200 motion embeddings in the input sequence and replace them with [MASK] tokens. The MAE head
201 aims to reconstruct the original VQ indices of these masked positions based on the contextual infor-
202 mation processed by the transformer encoder. Eq. (9) shows how for each masked position i , the
203 reconstruction is performed:

$$\hat{q}_i = \text{softmax}(W_{mae}h_i^* + b_{mae}), \quad (9)$$

204 where h_i^* is the transformer output at position i , and W_{mae} and b_{mae} are learnable parameters. The
205 MAE objective in Eq. (10) is formulated as a cross-entropy loss between the predicted and true VQ
206 indices:

$$\mathcal{L}_{mae} = -\frac{1}{|M|} \sum_{i \in M} \log \hat{q}_i[q_i], \quad (10)$$

207 where M is the set of masked positions and $\hat{q}_i[q_i]$ is the probability assigned to the true index q_i .

208 **Classification (CLS) Head.** For downstream activity recognition tasks, we utilize a classification
209 head that operates exclusively on the transformed [CLS] token representation. The [CLS] token ag-
210 gregates information from the entire sequence through self-attention mechanisms in the transformer
211 encoder. Eq. (11) defines how the classification is performed:

$$\hat{y} = \text{softmax}(W_{cls}h_{cls}^* + b_{cls}), \quad (11)$$

212 where h_{cls}^* is the transformed [CLS] token representation and W_{cls} and b_{cls} are learnable parameters.
213 The classification objective is formulated as a cross-entropy loss in Eq. (12):

$$\mathcal{L}_{cls} = -\sum_{j=1}^C y_j \log \hat{y}_j, \quad (12)$$

214 where y is the true activity label. During pre-training, we primarily focus on the MAE task, while
215 during different training phases, we balance various objectives through Eq. (13):

$$\mathcal{L} = \lambda_{mae}\mathcal{L}_{mae} + \lambda_{cls}\mathcal{L}_{cls} + \lambda_{vq}\mathcal{L}_{vq}, \quad (13)$$

216 where λ_{mae} , λ_{cls} , and λ_{vq} are hyperparameters balancing the different loss components from
217 Eqs. (10), (12), and (3), respectively. By adjusting these hyperparameters, we can control which
218 tasks are optimized during different training stages.

219 For fine-tuning on downstream tasks, we freeze the Motion Primitive Module and Context-Aware
220 Embedding Module while only updating the classification head parameters, enabling efficient adap-
221 tation to new datasets with minimal computational overhead.

222 4 Experiments

223 4.1 Experimental Setup

224 **Datasets.** We conducted extensive evaluations using six publicly available benchmark datasets:
225 PAMAP2 [38, 39], DSADS [7, 1], MHealth [5, 6], Realworld [46], UCI-HAR [40, 2], and USC-
226 HAD [56]. To ensure fair comparison with existing self-supervised methods, we pre-trained our
227 model on five of these datasets while using the remaining dataset for evaluation, maintaining consis-
228 tent experimental settings with other self-supervised approaches. All datasets are resampled to 100
229 Hz and segmented with a 500-sample window. To eliminate information leakage from overlapping
230 windows, we adopt the setting by using strides equal to the window size for all labeled training
231 segments. This strict non-overlapping approach may reduce absolute performance compared with
232 more permissive settings, but ensures a more rigorous evaluation. We use accuracy and macro-F1 as
233 evaluation metrics to assess model performance.

Table 1: Comparison with supervised pretrain-and-transfer baselines (upper block) and supervised train-from-scratch baselines (middle block) across six HAR datasets. Accuracy and macro-F1 are reported in percent; the rightmost column lists the mean over all datasets.

Method		PAMAP2	DSADS	MHealth	Realworld	UCI-HAR	USC-HAD	Average
BYOL [21]+perm_jit	Acc	80.88	91.30	88.94	85.18	82.66	74.04	83.83
	F1	79.01	89.22	88.59	86.81	80.32	73.92	82.98
BYOL [21]+lfc	Acc	76.76	90.80	89.74	86.08	81.18	72.82	82.90
	F1	74.64	89.39	88.84	85.99	80.87	69.23	81.49
BYOL [21]+Mixup	Acc	82.38	93.87	90.06	88.80	85.31	75.59	86.00
	F1	80.93	93.29	89.32	87.95	86.78	70.04	84.72
ModCL [22]	Acc	82.49	92.46	90.19	89.69	89.79	76.64	86.88
	F1	80.63	91.09	<u>89.98</u>	90.43	90.15	73.61	85.98
TSLANet [17]	Acc	86.76	98.12	89.84	90.03	91.02	<u>79.93</u>	<u>89.28</u>
	F1	<u>84.08</u>	97.43	87.85	<u>91.81</u>	<u>89.84</u>	77.74	<u>88.13</u>
CALANet [34]	Acc	85.10	96.01	86.05	90.07	88.38	77.37	87.16
	F1	83.94	95.58	83.80	91.59	86.91	72.36	85.70
MoPFormer	Acc	<u>86.08</u>	<u>97.60</u>	93.22	92.05	<u>90.58</u>	81.46	90.17
	F1	84.83	<u>97.38</u>	92.28	93.25	<u>89.98</u>	<u>77.67</u>	89.23
w/o pretrain	Acc	85.76	94.68	<u>90.53</u>	<u>91.84</u>	77.58	78.30	86.45
	F1	83.94	94.64	89.95	91.33	76.79	74.86	85.25

Training Protocol. All experiments were conducted on a Quadro RTX 8000 GPU. Our training protocol followed a two-stage approach. In the first pre-training stage, we trained the model across multiple datasets to extract a diverse set of motion primitives. We set the segment size to 50 samples for Motion Primitive length, used an internal embedding dimension of 256, and masked 25% of motion primitives as prediction targets for the MAE task. In the second stage, the classification head and transformer layers are fine-tuned on each downstream dataset.

Metadata Embedding Adapter. Sensor descriptors are standardized to a consistent string format (e.g., “body_part: Chest, sensor: acc, axis: x”). We then obtained embeddings with Google’s text-embedding-004 API, which enables the model to learn from contextual sensor metadata effectively.

4.2 Results

Baseline Comparisons. We compared MoPFormer with six different baseline approaches. Following the comprehensive evaluation in [35], we selected the BYOL framework [21], which demonstrated the best average performance among various contrastive learning frameworks. For time-domain and frequency-domain augmentations, we used perm_jit and lfc, respectively, which showed superior performance in their domains. As noted in [15], Mixup provides distinctive benefits compared to other temporal augmentation methods, so we included it as a baseline. ModCL [22] represents a state-of-the-art contrastive learning method that leverages both intra-modal and inter-modal consistency in wearable sensor data. We included it as a contemporary contrastive learning approach. For all contrastive learning methods, we maintained a consistent 0.2:0.8 train-test split ratio.

Additionally, we compared against recent supervised approaches, including TSLANet [17] (ICML 2024), a widely recognized foundational model in the time-series domain, and CALANet [34] (NeurIPS 2024), which was specifically designed for HAR tasks [8]. Detailed parameter configurations for all baseline models are provided in Appendix A.

Performance Analysis. Tab. 1 presents comprehensive results across all six datasets. MoPFormer achieves state-of-the-art performance, obtaining the highest average accuracy and F1 scores. Specifically, our model achieved the best performance on MHealth (93.22% Accuracy, 92.28% F1) and USC-HAD (81.46% Accuracy), while maintaining competitive results on other datasets.

The ablation study (w/o pretrain) demonstrates that our pretraining strategy significantly contributes to model performance, particularly on datasets like UCI-HAR with limited data but complex activity patterns, where performance drops by 13 percentage points without pretraining. This confirms the value of our two-stage training approach with motion primitive extraction and masked autoencoding.

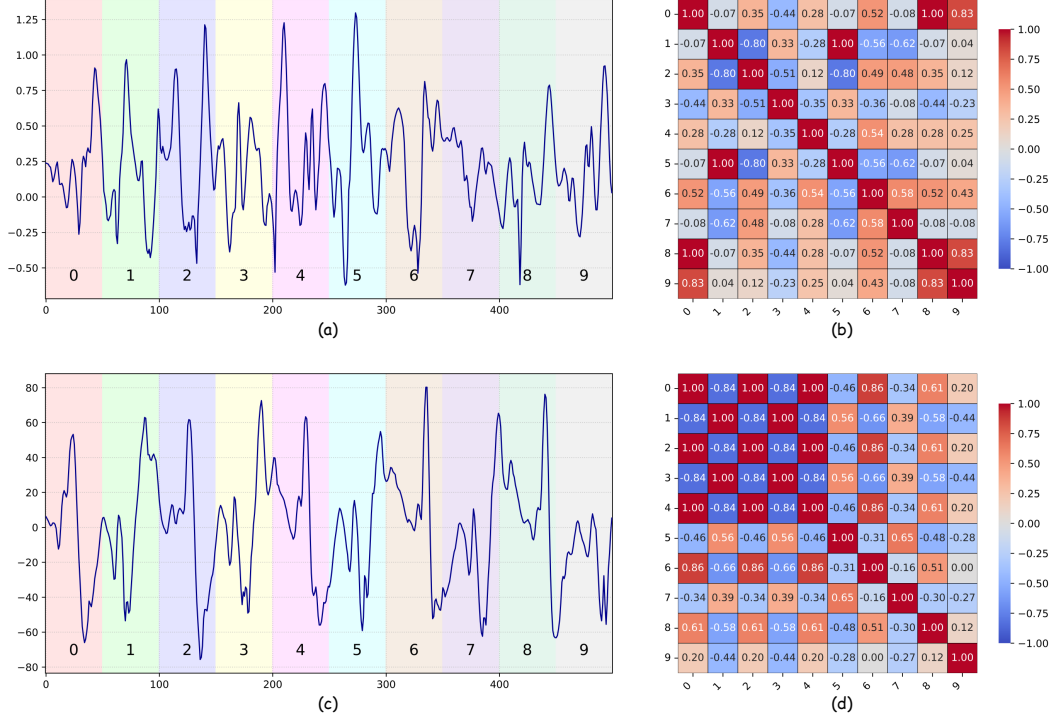


Figure 3: Motion primitive segmentation and similarity analysis. (a) 5-second raw accelerometer trace from USC-HAD dataset, segmented into ten 0.5-second motion primitives. (b) Cosine-similarity matrix of motion primitive embeddings from accelerometer data. (c) Corresponding 5-second gyroscope trace with identical segmentation. (d) Cosine-similarity matrix of embeddings for gyroscope-based motion primitives. The matrices reveal pattern correlations between different motion primitives after Motion Primitive Embedding processing.

Ablation Study. To validate each component’s contribution, we conduct an ablation study on two representative datasets (Tab. 2). First, removing the pretraining stage leads to a modest accuracy drop, highlighting the importance of motion-primitive initialization in cross-modal generalization. Next, further omitting the statistical features or excluding metadata embedding causes a dramatic performance loss. Together, these findings show that pretraining, statistical features, and metadata embeddings each deliver unique information, and that the combination is critical to MoPFormer’s superior performance. More results are put in Appendix B.

MoPFormer’s strong performance across diverse datasets validates our motion-centric approach and demonstrates its generalization capabilities. In the following section, we will further analyze the extracted motion primitives better to understand their contribution to the model’s effectiveness.

5 Analysis

5.1 Motion Primitive Similarity

To examine what the VQ codebook captures, we analyzed two example segments of five seconds each, shown in Fig. 3. Each trace is divided into ten 0.5s motion primitives, and we compute pairwise cosine similarity between the embeddings of those motion primitives. A consistent pattern emerges: primitives describing similar kinematic shapes, such as indices 8 and 9 in (a) or indices 1 and 3 in

Table 2: Ablation study for each component on the PAMAP2 and DSADS datasets.

Method	PAMAP2		DSADS	
	Acc	F1	Acc	F1
MoPFormer	86.08	84.83	97.60	97.38
- w/o Pretrain	85.76	83.94	94.68	94.64
- w/o Statistical Feature	69.95	53.60	71.15	62.34
- w/o Metadata Embedding	58.18	53.10	80.94	75.40

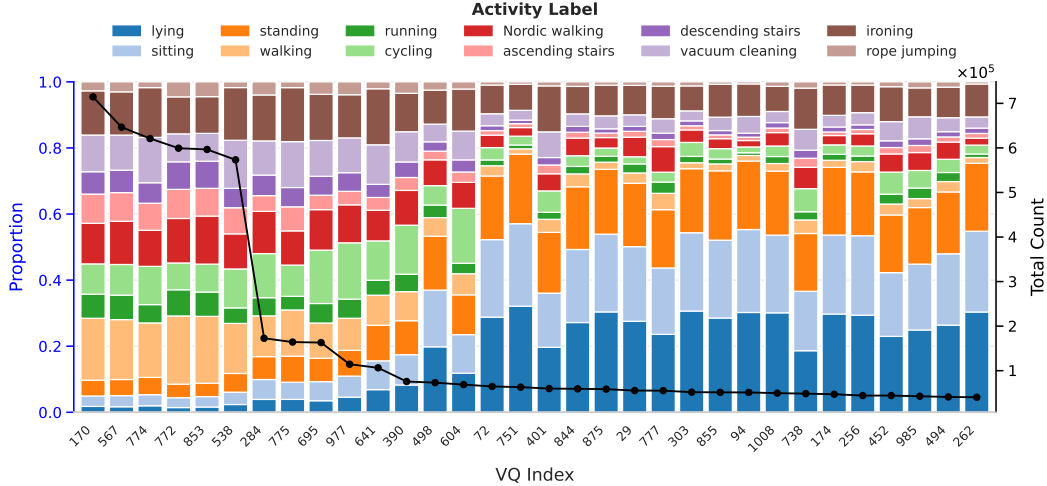


Figure 4: Frequency and activity composition of the 32 most common motion primitives from PAMAP2. The stacked bars (left axis) show the proportion of each activity label for every VQ index, while the black line (right axis) plots the absolute occurrence count of motion primitives.

(c), cluster into blocks with high cosine similarity. In contrast, primitives with opposite or dissimilar trends (e.g., indices 1 and 3 in (a), and indices 0 and 1 in (c)) exhibit low similarity.

5.2 Motion Primitive Frequency

To further explain the rationale of the learned codebook, we analyzed both the overall frequency of each primitive and its distribution across activity classes (using the PAMAP2 dataset for illustration). Fig. 4 presents the 32 most frequent VQ indices: each stacked bar depicts the relative share of 12 activity classes, and the black curve shows the absolute number of occurrences. Because any macroscopic activity can be decomposed into a sequence of fine-grained motion primitives, the codebook naturally captures micro-movements that are reused across different activities. The resulting distribution is highly skewed: the ten most common primitives cover the vast majority of windows, whereas the remaining indices form a long-tailed set that represents rare or transitional motions. Notably, locomotion classes (e.g., walking, running, and cycling) dominate the high-frequency primitives (e.g., primitive #170 and primitive #567), while low-variance postures such as lying, sitting, and standing are scattered among the less frequent primitives (e.g., primitive #72 and primitive #751). This frequency analysis could allow us to identify which primitive motions are more common and how they relate to high-level activities, which is a step toward explaining model decisions.

6 Conclusion and Future Work

In this work, we introduced MoPFormer, a motion-primitives-based Transformer framework for wearable-sensor activity recognition. Our approach tackles two key HAR challenges: interpretability and cross-domain generalization. Through comprehensive experiments, we showed that MoPFormer achieves state-of-the-art classification performance on six diverse datasets, outperforming both supervised and self-supervised baselines. We also demonstrated that MoPFormers learned primitives are semantically meaningful. This interpretability lets us peek inside the “black box” of the HAR model. Overall, MoPFormer combines the strengths of sequence modeling and symbolic representation learning, yielding a HAR system that is both accurate and explainable.

Our future work will focus on two key directions. First, developing more robust foundational models by pre-training on larger, more diverse sensor datasets to extract richer motion primitives. Second, aligning our learned motion primitive embeddings with large language models as a novel input modality enables more flexible activity recognition and potentially expresses transfer ability.

References

- [1] K. Altun, B. Barshan, and O. Tunçel. “Comparative study on classifying human activities with miniature inertial and magnetic sensors”. In: *Pattern Recognit.* 43 (2010), pp. 3605–3620. URL: <https://api.semanticscholar.org/CorpusID:18488847>.
- [2] D. Anguita et al. “A Public Domain Dataset for Human Activity Recognition using Smart-phones”. In: *The European Symposium on Artificial Neural Networks*. 2013. URL: <https://api.semanticscholar.org/CorpusID:6975432>.
- [3] S. Arabzadeh, F. Almasganj, and M. M. Ahmadi. “CNN Autoencoders for Hierarchical Feature Extraction and Fusion in Multi-sensor Human Activity Recognition”. In: *arXiv preprint arXiv:2502.04489* (2025).
- [4] M. H. Arshad, M. Bilal, and A. Gani. “Human activity recognition: Review, taxonomy and open challenges”. In: *Sensors* 22.17 (2022), p. 6463.
- [5] O. Banos, R. Garcia, and A. Saez. *MHEALTH*. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C5TW22>. 2014.
- [6] O. Banos et al. “Design, implementation and validation of a novel open framework for agile development of mobile health applications”. In: *Biomedical engineering online* 14 (2015), pp. 1–20.
- [7] B. Barshan and K. Altun. *Daily and Sports Activities*. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C5C59F>. 2010.
- [8] C. Bergmeir. *Fundamental limitations of foundational forecasting models: The need for multimodality and rigorous evaluation*. Talk at the Conference on Neural Information Processing Systems (NeurIPS 2024). Accessed: 2025-05-07. Dec. 2024. URL: <https://cbergmeir.com/talks/neurips2024/>.
- [9] O. Celik, A. Taranovic, and G. Neumann. “Acquiring diverse skills using curriculum reinforcement learning with mixture of experts”. In: *Proceedings of the 41st International Conference on Machine Learning*. ICML’24. Vienna, Austria: JMLR.org, 2024.
- [10] B. Chen et al. “Comodo: Cross-modal video-to-imu distillation for efficient egocentric human activity recognition”. In: *arXiv preprint arXiv:2503.07259* (2025).
- [11] K. Chen et al. “Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities”. In: *ACM Computing Surveys (CSUR)* 54.4 (2021), pp. 1–40.
- [12] W. Cheng et al. “Predicting Complex Activities from Ongoing Multivariate Time Series”. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, July 2018, pp. 3322–3328. DOI: 10.24963/ijcai.2018/461. URL: <https://doi.org/10.24963/ijcai.2018/461>.
- [13] R. R. Chowdhury et al. “Tarnet: Task-aware reconstruction for time-series transformer”. In: *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data Mining*. 2022, pp. 212–220.
- [14] A. Doherty et al. “Large scale population assessment of physical activity using wrist worn accelerometers: the UK biobank study”. In: *PloS one* 12.2 (2017), e0169649.
- [15] E. Eldele et al. “Contrastive Domain Adaptation for Time-Series Via Temporal Mixup”. In: *IEEE Transactions on Artificial Intelligence* 5.3 (2024), pp. 1185–1194. DOI: 10.1109/TAI.2023.3293473.
- [16] E. Eldele et al. “Time-series representation learning via temporal and contextual contrasting”. In: *arXiv preprint arXiv:2106.14112* (2021).
- [17] E. Eldele et al. “TSLANet: rethinking transformers for time series representation learning”. In: *Proceedings of the 41st International Conference on Machine Learning*. ICML’24. Vienna, Austria: JMLR.org, 2024.
- [18] M. Al-Faris et al. “A review on computer vision-based methods for human action recognition”. In: *Journal of imaging* 6.6 (2020), p. 46.
- [19] D. Geissler, B. Zhou, and P. Lukowicz. “Strategies and Challenges of Efficient White-Box Training for Human Activity Recognition”. In: *arXiv preprint arXiv:2412.08507* (2024).
- [20] A. Grattafiori et al. “The llama 3 herd of models”. In: *arXiv preprint arXiv:2407.21783* (2024).

- [21] J.-B. Grill et al. “Bootstrap your own latent-a new approach to self-supervised learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 21271–21284.
- [22] C. Guo et al. “Modality Consistency-Guided Contrastive Learning for Wearable-Based Human Activity Recognition”. In: *IEEE Internet of Things Journal* 11.12 (2024), pp. 21750–21762. DOI: 10.1109/JIOT.2024.3379019.
- [23] Z. Hussain, M. Sheng, and W. E. Zhang. “Different approaches for human activity recognition: A survey”. In: *arXiv preprint arXiv:1906.05074* (2019).
- [24] J. V. Jeyakumar et al. “SenseHAR: a robust virtual activity sensor for smartphones and wearables”. In: *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. SenSys ’19. New York, New York: Association for Computing Machinery, 2019, pp. 15–28. ISBN: 9781450369503. DOI: 10.1145/3356250.3360032. URL: <https://doi.org/10.1145/3356250.3360032>.
- [25] A. Kamboj and M. Do. “A Survey of IMU Based Cross-Modal Transfer Learning in Human Activity Recognition”. In: *arXiv preprint arXiv:2403.15444* (2024).
- [26] B. Li et al. “Two-stream convolution augmented transformer for human activity recognition”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. 1. 2021, pp. 286–293.
- [27] Z. C. Lipton. “The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.” In: *Queue* 16.3 (2018), pp. 31–57.
- [28] S. Liu et al. “Focal: Contrastive learning for multimodal time-series sensing signals in factorized orthogonal latent space”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 47309–47338.
- [29] H. Ma et al. “AttnSense: Multi-level attention mechanism for multimodal human activity recognition.” In: *IJCAI*. 2019, pp. 3109–3115.
- [30] F. J. Ordóñez and D. Roggen. “Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition”. In: *Sensors* 16.1 (2016), p. 115.
- [31] A. Padmanabha et al. “EgoCHARM: Resource-Efficient Hierarchical Activity Recognition using an Egocentric IMU Sensor”. In: *arXiv preprint arXiv:2504.17735* (2025).
- [32] G. Panahandeh et al. “Continuous hidden Markov model for pedestrian activity classification and gait analysis”. In: *IEEE Transactions on Instrumentation and Measurement* 62.5 (2013), pp. 1073–1083.
- [33] J. Park, D.-W. Kim, and J. Lee. “CALANet: Cheap All-Layer Aggregation for Human Activity Recognition”. In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: <https://openreview.net/forum?id=ouoBW2PXFQ>.
- [34] J. Park, D.-W. Kim, and J. Lee. “CALANet: Cheap All-Layer Aggregation for Human Activity Recognition”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Globerson et al. Vol. 37. Curran Associates, Inc., 2024, pp. 69419–69444. URL: https://proceedings.neurips.cc/paper_files/paper/2024/file/8078e76f913e31b8467e85b4c0f0d22b-Paper-Conference.pdf.
- [35] H. Qian, T. Tian, and C. Miao. “What Makes Good Contrastive Learning on Small-Scale Wearable-based Tasks?” In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. KDD ’22. Washington DC, USA: Association for Computing Machinery, 2022, pp. 3761–3771. ISBN: 9781450393850. DOI: 10.1145/3534678.3539134. URL: <https://doi.org/10.1145/3534678.3539134>.
- [36] A. Radford et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PmlR. 2021, pp. 8748–8763.
- [37] V. Radu et al. “Multimodal Deep Learning for Activity and Context Recognition”. In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1.4 (Jan. 2018). DOI: 10.1145/3161174. URL: <https://doi.org/10.1145/3161174>.
- [38] A. Reiss. *PAMAP2 Physical Activity Monitoring*. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C5NW2H>. 2012.
- [39] A. Reiss and D. Stricker. “Introducing a New Benchmarked Dataset for Activity Monitoring”. In: *2012 16th International Symposium on Wearable Computers* (2012), pp. 108–109. URL: <https://api.semanticscholar.org/CorpusID:10337279>.
- [40] J. L. Reyes-Ortiz et al. *Human Activity Recognition Using Smartphones*. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C54S4K>. 2013.

- [41] M. Sanzari, V. Ntouskos, and F. Pirri. “Discovery and recognition of motion primitives in human activities”. In: *PloS one* 14.4 (2019), e0214499.
- [42] N. sedaghati, S. ardebili, and A. Ghaffari. “Application of human activity/action recognition: a review”. In: *Multimedia Tools and Applications* (2025), pp. 1–30.
- [43] S. Serrano and N. A. Smith. “Is attention interpretable?” In: *arXiv preprint arXiv:1906.03731* (2019).
- [44] Y.-H. Shen, K.-X. He, and W.-Q. Zhang. “SAM-GCNN: a gated convolutional neural network with segment-level attention mechanism for home activity monitoring”. In: *2018 IEEE international symposium on signal processing and information technology (ISSPIT)*. IEEE. 2018, pp. 679–684.
- [45] D. Spathis et al. “Self-supervised transfer learning of physiological representations from free-living wearable data”. In: *Proceedings of the Conference on Health, Inference, and Learning*. 2021, pp. 69–78.
- [46] T. Szttyler 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)*. 2016, pp. 1–9. DOI: 10.1109/PERCOM.2016.7456521.
- [47] A. Van Den Oord, O. Vinyals, et al. “Neural discrete representation learning”. In: *Advances in neural information processing systems* 30 (2017).
- [48] A. Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).
- [49] H. Wu et al. “Timesnet: Temporal 2d-variation modeling for general time series analysis”. In: *arXiv preprint arXiv:2210.02186* (2022).
- [50] C. Xu et al. “A shallow convolution network based contextual attention for human activity recognition”. In: *International conference on mobile and ubiquitous systems: Computing, networking, and services*. Springer. 2022, pp. 155–171.
- [51] C. Xu et al. “An Enhanced Human Activity Recognition Algorithm with Positional Attention”. In: *Proceedings of The 14th Asian Conference on Machine Learning*. Ed. by E. Khan and M. Gonen. Vol. 189. Proceedings of Machine Learning Research. PMLR, Dec. 2023, pp. 1181–1196. URL: <https://proceedings.mlr.press/v189/xu23a.html>.
- [52] Z. Xu, A. Zeng, and Q. Xu. “FITS: Modeling time series with 10k parameters”. In: *arXiv preprint arXiv:2307.03756* (2023).
- [53] Z. Yue et al. “Ts2vec: Towards universal representation of time series”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 36. 8. 2022, pp. 8980–8987.
- [54] M. Zeng et al. “Understanding and improving recurrent networks for human activity recognition by continuous attention”. In: *Proceedings of the 2018 ACM international symposium on wearable computers*. 2018, pp. 56–63.
- [55] G. Zerveas et al. “A transformer-based framework for multivariate time series representation learning”. In: *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. 2021, pp. 2114–2124.
- [56] M. Zhang and A. A. Sawchuk. “USC-HAD: A Daily Activity Dataset for Ubiquitous Activity Recognition Using Wearable Sensors”. In: *ACM International Conference on Ubiquitous Computing (Ubicomp) Workshop on Situation, Activity and Goal Awareness (SAGAware)*. Pittsburgh, Pennsylvania, USA, Sept. 2012.

A Experimental Settings

In this section, we provide detailed parameter settings for our proposed MoPFormer model and baseline models used in our experiments. All experiments can run normally on the RTX 8000 with 40GB VRAM (requiring the use of gradient accumulation techniques).

A.1 MoPFormer Parameter Settings

We implement the MoPFormer model with the following configuration. For motion representation, we use 50 as the motion primitive length, which corresponds to 0.5s of motion when resampled to 100Hz. The codebook size is set to 1024 with an embedding dimension of 256. The architecture consists of 5 standard Transformer Encoder layers with 8 attention heads per layer and an MLP ratio of 1. We use GELU as the activation function throughout the network. Notably, we do not employ any dropout in our model. During training, we apply a masking ratio of 0.25 for the masked modeling objective. For optimization, we employ the AdamW optimizer with a learning rate of $1e-4$ and a weight decay coefficient of $1e-5$. We use a batch size of 512, implemented with gradient accumulation using PyTorch Lightning to optimize memory usage.

A.2 Baseline Model Parameter Settings

BYOL For BYOL [21] (Bootstrap Your Own Latent), we implement the model following the CL-HAR [35] library’s configuration. The architecture consists of two networks: an online network and a target network, both utilizing the DCL (DeepConvLSTM) architecture as the backbone encoder. The online network includes a projection head that maps the backbone’s output to a 128-dimensional embedding space with a hidden layer of the same dimension, structured as a sequence of Linear, BatchNorm1d, ReLU, Linear, and BatchNorm1d layers. The prediction head follows with the same dimensionality (128) and is structured as Linear, BatchNorm1d, ReLU, and Linear layers. The target network’s parameters are updated via an exponential moving average of the online network’s parameters with a decay rate of 0.996. We employ three different data augmentation techniques, all following the settings provided in [35]. The model is trained using negative cosine similarity as the loss function with the Adam optimizer. We set the learning rate to $1e-4$ for the online encoder and $1e-3$ for the online predictor, with a weight decay of $1.5e-6$, using a CosineAnnealingLR scheduler. For downstream activity recognition, we freeze the backbone encoder and train a linear classifier with the Adam optimizer at a learning rate of $1e-3$ for the same number of epochs.

ModCL For ModCL, we follow the default parameter settings as described in the original paper. Since the official code was not publicly available, we implemented the model based on the paper’s specifications. During our reproduction, we enabled several data augmentation techniques, specifically applying five methods: Jittering, Scaling, Permutation, Masking, and Time Warping, all using the same proportions and parameters as specified in the original work.

TSLANet For TSLANet, we implement the model using the default parameter settings specified in the original paper. For parameters not explicitly mentioned in the paper, we adopt the default configurations from the authors’ publicly released code repository. Following the original architecture, we enable both the ICB (Input Channel Block) and ASB (Attention Sub-Block) modules. The model is trained with the same optimization strategy and hyperparameters as described in the original TSLANet paper.

CALANet For CALANet, we follow the default parameter settings described in the original paper. For configuration details not explicitly stated in the paper, we use the default settings provided in the authors’ official code implementation. We set the number of layers in the layer aggregation pool to 9 based on the authors’ ablation study results, which identified this as the optimal configuration. All other hyperparameters and training procedures follow the specifications in the original paper.

B Supplementary Ablation Studies

Since our work primarily focuses on investigating interpretable motion primitives in human activity recognition (HAR), the main paper demonstrates the necessity of each architectural component

through module-wise ablation experiments. This appendix provides additional ablation studies examining the impact of key hyperparameters on model performance to provide comprehensive insights into our method’s design choices.

B.1 Motion Primitive Window Length

The choice of motion primitive window length significantly affects the model’s ability to capture meaningful motion patterns. We evaluate different window lengths to determine the optimal temporal granularity for motion primitive extraction.

Table 3: Ablation study on motion primitive window length

Window Length	PAMAP2		DSADS	
	Acc	F1	Acc	F1
25	85.71	82.11	97.09	96.37
50 (ours)	86.08	84.83	97.60	97.38
100	80.81	79.85	92.94	92.17

As shown in Table 3, a window length of 50 achieves optimal performance across both datasets. While shorter windows 25 demonstrate comparable performance, we select the length of 50 to ensure sufficient temporal context for comprehensive motion primitive analysis. This choice prioritizes capturing complete motion phases within each primitive, which is crucial for meaningful interpretability analysis. Longer windows 100 lead to substantial performance degradation due to the inclusion of multiple distinct motion phases within a single primitive, confirming that our selected window length provides the appropriate temporal granularity without compromising motion primitive coherence.

B.2 Transformer Encoder Depth

We investigate the impact of transformer encoder depth on the model’s representational capacity and its effect on motion primitive quality.

Table 4: Ablation study on transformer encoder layer number

Depth	PAMAP2		DSADS	
	Acc	F1	Acc	F1
2	85.01	84.59	96.92	96.56
5 (ours)	86.08	84.83	97.60	97.38
7	86.11	84.98	97.53	97.31
10	86.05	84.79	97.28	97.02

The results in Table 4 show that performance differences between 5, 7, and 10 layers are minimal. However, we adopt 5 layers as our configuration to ensure adequate modeling capacity for motion primitive extraction without introducing unnecessary complexity. This choice reflects our preference for slightly conservative parameter settings that guarantee sufficient representational power for motion primitive analysis, while avoiding potential overfitting. The 2-layer configuration shows reduced performance, confirming that our selected depth provides the necessary capacity for high-quality motion primitive learning.

B.3 Hidden Embedding Dimensionality

The dimensionality of hidden embeddings critically determines the model’s representational capacity for motion primitive encoding and subsequent interpretability analysis.

From Table 5, we observe that performance reaches saturation around 128 dimensions, with 256 dimensions providing marginal improvements. We choose 256 dimensions to ensure comprehensive representational capacity that does not limit our motion primitive analysis. This decision follows

Table 5: Ablation study on hidden embedding size

Embedding Size	PAMAP2		DSADS	
	Acc	F1	Acc	F1
32	82.18	81.70	94.81	93.88
64	85.90	85.86	96.93	96.31
128	86.45	85.50	97.28	97.35
256 (ours)	86.08	84.83	97.60	97.38

our principle of using slightly excessive parameters to guarantee that motion primitive quality and interpretability are not compromised by insufficient representational capacity. Lower dimensions (32-64) show significant performance degradation, validating that our selected dimensionality provides adequate space for capturing the complexity and nuances of human motion patterns.

B.4 Vector Quantization Codebook Size

The VQ codebook size determines the discrete vocabulary available for motion primitive representation, directly impacting both model performance and the richness of discoverable motion primitives.

Table 6: Ablation study on VQ codebook size

Codebook Size	PAMAP2		DSADS	
	Acc	F1	Acc	F1
16	78.81	76.89	90.05	88.93
64	84.31	83.79	95.55	95.43
256	85.62	84.87	97.52	97.18
1024 (ours)	86.08	84.83	97.60	97.38

As demonstrated in Table 6, performance steadily improves with larger codebook sizes and appears to saturate around 256 codes for the current pretraining datasets. However, we adopt 1024 codes to ensure comprehensive coverage of motion primitive diversity, particularly to avoid any limitations that might affect our detailed motion primitive analysis. This choice exemplifies our approach of using generously-sized parameters to guarantee that the learned motion primitives fully capture the richness and subtlety of human motion patterns without being constrained by codebook capacity. Smaller codebooks (16-64) exhibit substantial performance degradation, confirming the necessity of adequate representational diversity for meaningful motion primitive discovery.

C Additional Motion Primitive Analysis

To provide deeper insights into the interpretability of our learned motion primitives, we analyze the distribution of motion primitive usage across different human activities. This analysis demonstrates how distinct activities exhibit characteristic motion primitive patterns, validating the semantic meaningfulness of our learned representations. Figures 5 through 12 present the motion primitive usage distributions for eight representative activities from our dataset.

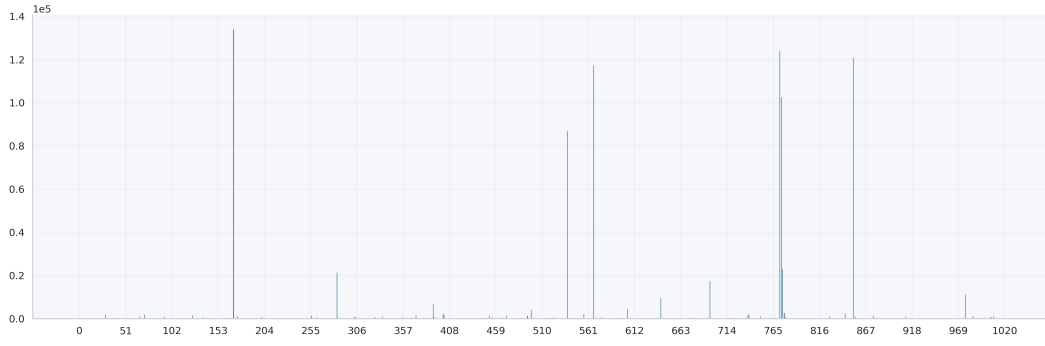


Figure 5: Motion primitive usage distribution for ascending stairs activity.

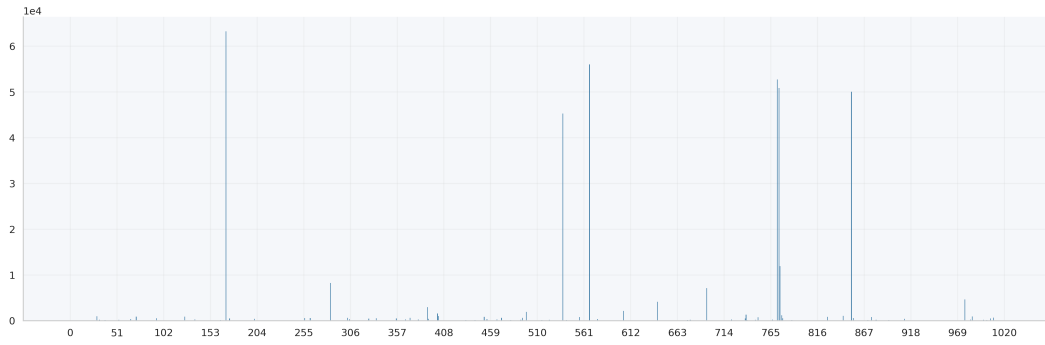


Figure 6: Motion primitive usage distribution for cycling activity.

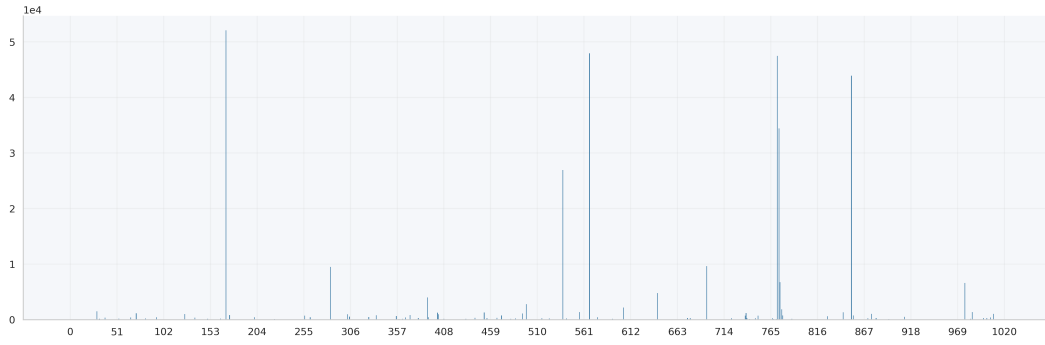


Figure 7: Motion primitive usage distribution for walking activity.

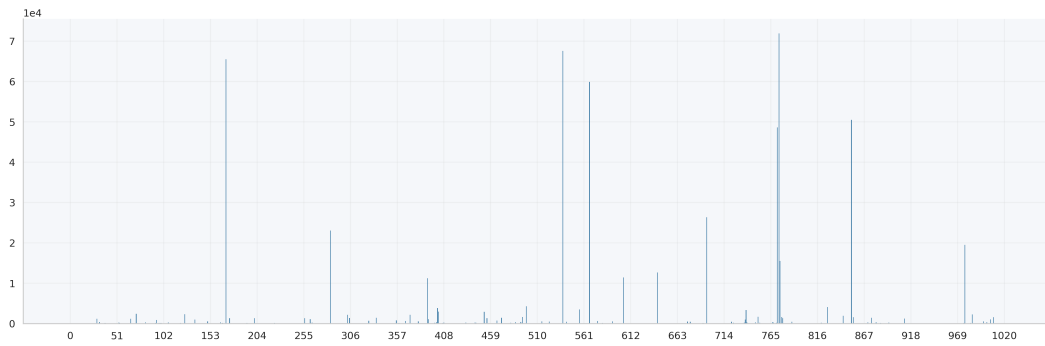


Figure 8: Motion primitive usage distribution for running activity.

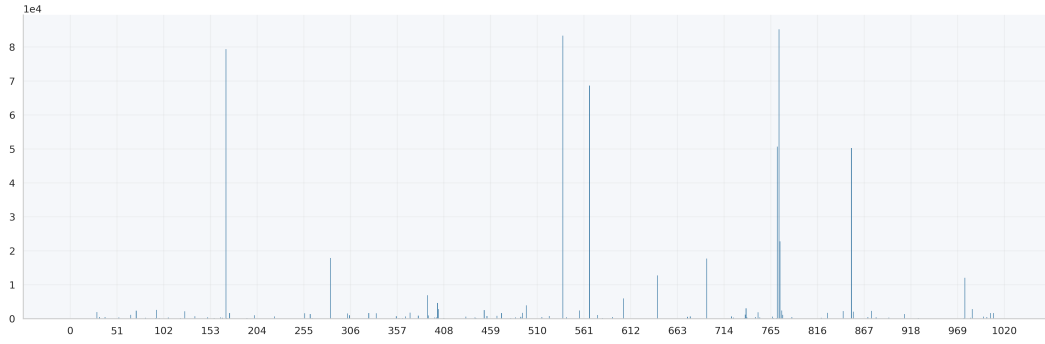


Figure 9: Motion primitive usage distribution for jumping activity.

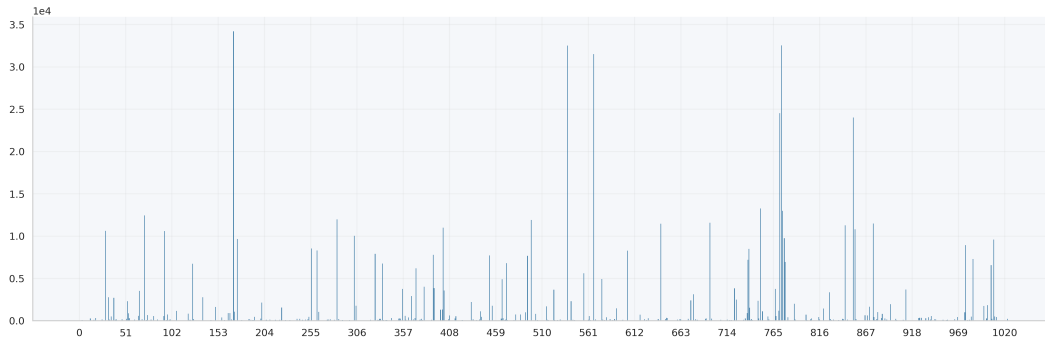


Figure 10: Motion primitive usage distribution for lying activity.

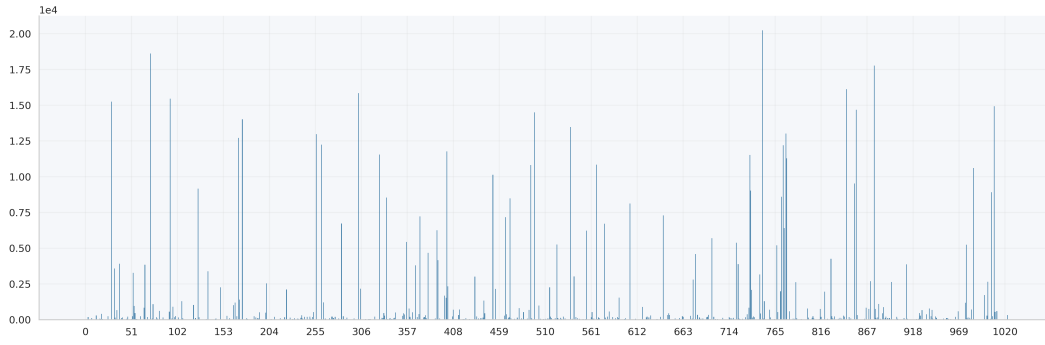


Figure 11: Motion primitive usage distribution for sitting activity.

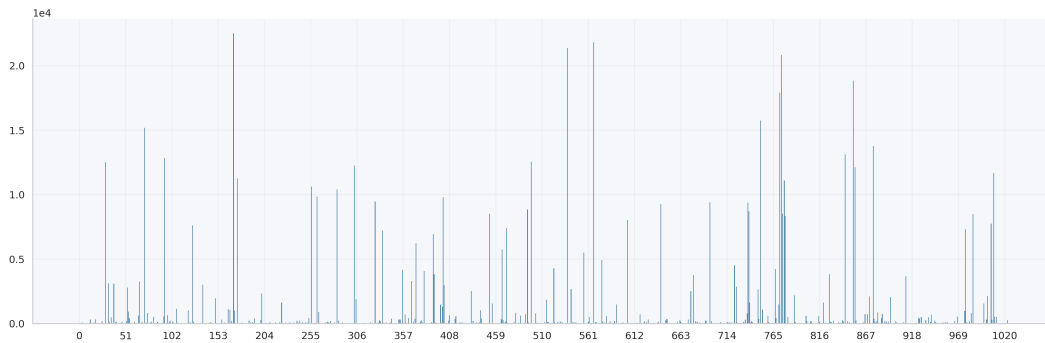


Figure 12: Motion primitive usage distribution for standing activity.

Through careful observation of these distributions, we can identify distinct patterns that reflect the fundamental differences in motion primitive composition across various activities. The distributions clearly separate into two categories: dynamic activities (Figures 5 through 9) and static activities (Figures 10 through 12). This contrast is particularly striking in terms of primitive usage patterns. Dynamic activities demonstrate concentrated utilization of specific motion primitives, with certain primitives exhibiting significantly higher activation frequencies, indicating the presence of characteristic movement patterns essential for these activities. In contrast, static activities show a more uniform distribution across motion primitives, suggesting these behaviors rely on a broader range of subtle motion components for postural maintenance and minor adjustments. This fundamental difference in primitive usage validates that our learned motion primitives capture meaningful biomechanical distinctions between different categories of human activities.

Furthermore, to analyze the temporal relationships between motion primitives, we construct and visualize the Markov transition probability matrix for all learned motion primitives, as illustrated in Figure 13.

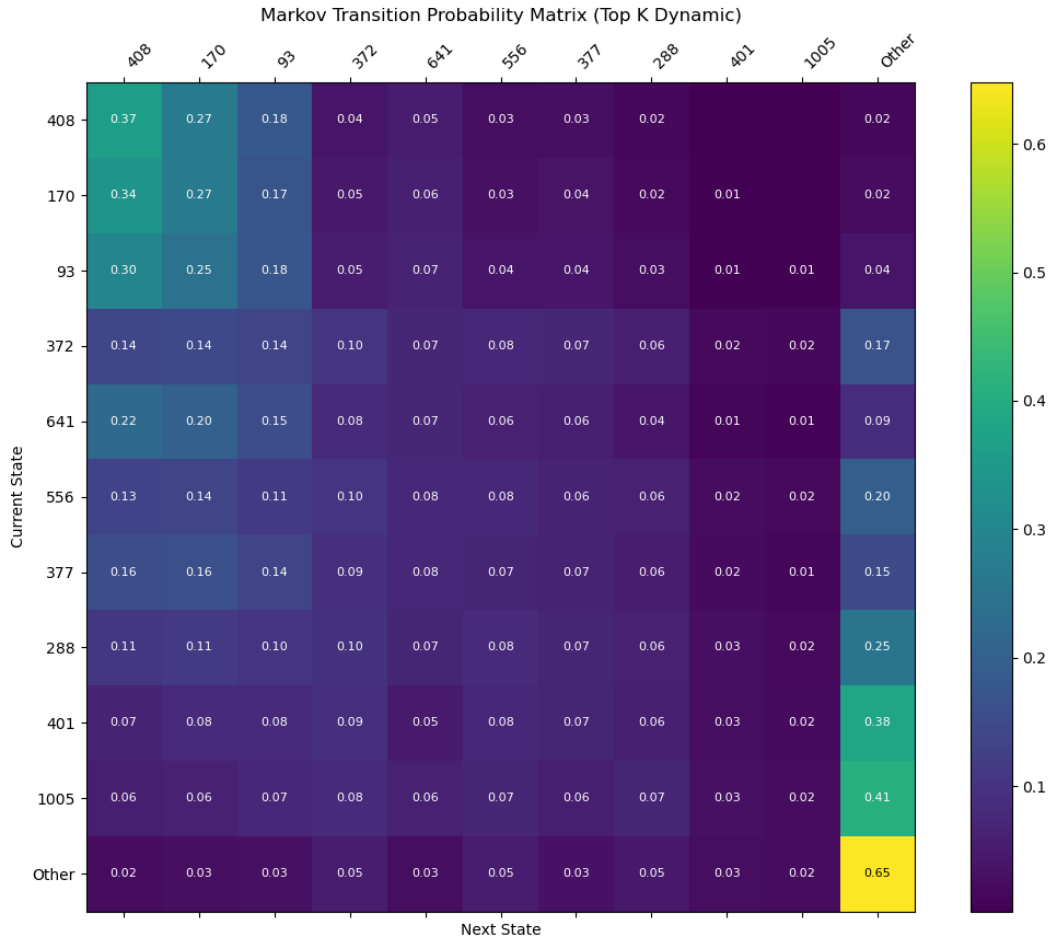


Figure 13: Markov transition probability matrix showing the temporal dependencies between motion primitives.

D Dataset Description

All datasets used in this paper adopt a sliding window approach with a window size of 500 samples and a step size of 500 samples as the basic recognition unit. All sensor data is resampled to 100 Hz to ensure consistency across different datasets. Data segments shorter than the specified window size are discarded from the analysis. For fine-tuning and evaluation purposes, we allocate 20% of

each dataset for fine-tuning while the remaining data is used for testing. The datasets employed in this study are described as follows:

PAMAP2 The PAMAP2 Physical Activity Monitoring dataset is a comprehensive collection designed for activity recognition and intensity estimation research, containing recordings of 18 different physical activities such as walking, cycling, and playing soccer performed by 9 subjects. Each subject wore three Colibri wireless inertial measurement units (IMUs) positioned on the wrist of the dominant arm, chest, and ankle of the dominant side, sampling at 100 Hz, along with a heart rate monitor operating at approximately 9 Hz. The data collection protocol required each subject to perform 12 standardized activities, with some subjects also completing additional optional activities.

DSADS This dataset captures activity recognition data from 19 different physical activities performed by 8 subjects (4 female, 4 male, aged 20-30). Activities include basic postures, locomotion, daily activities, and exercises, each performed for 5 minutes and segmented into 5-second intervals. Five sensor units were placed on different body locations (torso, arms, legs), with each unit containing accelerometer, gyroscope, and magnetometer sensors. Data was recorded at 25 Hz and organized hierarchically by activity, subject, and segment.

MHealth This dataset contains human activity recognition data from 12 physical activities (stationary postures, locomotion, and exercises) performed by 10 volunteers in natural settings. Three wearable sensors placed on the chest, right wrist, and left ankle measure motion parameters (acceleration, rotation, magnetic orientation), with the chest sensor also capturing ECG data. All recordings were sampled at 50 Hz, providing comprehensive movement and physiological data for activity recognition research.

Realworld This dataset captures human activity recognition data from 8 different physical activities (walking, running, sitting, standing, lying, stairs up, stairs down, jumping) performed by 15 subjects (8 males, 7 females; age 31.9 ± 12.4 , height 173.1 ± 6.9 cm, weight 74.1 ± 13.8 kg). Each activity was performed for approximately 10 minutes per subject (except jumping at ~ 1.7 minutes due to physical exertion), with data equally distributed between genders. The dataset includes IMU sensor readings (acceleration, gyroscope, magnetic field) collected at a 50 Hz sampling rate simultaneously from 7 different body positions (chest, forearm, head, shin, thigh, upper arm, and waist).

UCI-HAR This dataset captures human activity recognition data from 6 different physical activities (walking, walking upstairs, walking downstairs, sitting, standing, lying) performed by 30 volunteers aged 19-48 years. The experiments were video-recorded for manual data labeling. The sensor signals were pre-processed with noise filters, while a Butterworth low-pass filter with 0.3 Hz cutoff frequency was applied to separate body acceleration from gravity, with features extracted from both time and frequency domains for activity recognition analysis.

USC-HAD This dataset captures human motion data using MotionNode sensors operating at 100Hz ($\pm 6g$ accelerometer range, $\pm 500dps$ gyroscope range). It includes 12 different physical activities: Walking Forward/Left/Right, Walking Upstairs/Downstairs, Running Forward, Jumping Up, Sitting, Standing, Sleeping, and Elevator Up/Down.

E Limitations

Our approach introduces computational overhead through Vector Quantization and multi-layer Transformer processing. This computational complexity could pose challenges for real-time inference on resource-limited wearable devices, potentially limiting its deployment in edge computing scenarios.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: The abstract and introduction clearly state the three main contributions: (1) the MoPFormer pre-training framework, (2) a representation method that enables unified training across heterogeneous datasets, and (3) extraction of motion primitives with detailed interpretability analysis. These claims are substantiated throughout the paper, particularly in the methodology, experiments, and analysis.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: See Appendix.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides comprehensive details on datasets (Section 4.1), preprocessing steps (resampling to 100 Hz, segmentation with 500-sample windows), architecture specifics (segment size, embedding dimensions, masking ratio), and training protocol. Additionally, the methodology section fully describes all model components, loss functions, and training objectives. Additionally, we will provide further details in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

- (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code will be provided soon.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so No is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The paper details all necessary details about the training and testing processes.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The paper does not report error bars or other statistical significance measures.

Guidelines:

- The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: See Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The research focuses on analyzing public, anonymized sensor datasets for activity recognition, which presents minimal ethical concerns. The methodology and evaluations follow standard practices in machine learning research without raising issues related to privacy, harm, or misrepresentation.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: As HAR is a fundamental technique in healthcare, fitness tracking, and surveillance, our accurate and real-time model can reduce the negative societal impacts due to false predictions.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [\[Yes\]](#)

Justification: We will include the usage guidelines or restrictions in code release.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [\[Yes\]](#)

Justification: We cited the original paper that produced the datasets and our source code release follows CC BY 4.0 license.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper and the code release include the details about training, license, usage guidelines, etc.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve any research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve any research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We only used an LLM for language polishing of the paper.

Guidelines:

- In the Metadata Embedding Adapter module, we used Google’s text-embedding-004 API to convert the original sensor information into embeddings.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.