# Transformer-based Dog Behavior Classification with Motion Sensors

Barak Or, *Member, IEEE*

***Abstract*—This paper deals with classifying dog behavior using motion sensors, leveraging a transformer-based Deep Neural Network (DNN) model. Understanding dog behavior is essential for fostering positive relationships between dogs and humans and ensuring their well-being. Traditional methods often fall short in capturing temporal dependencies and efficiently processing high-dimensional sensor data. Our proposed architecture, inspired by its success in Natural Language Processing (NLP), utilizes the self-attention mechanism of the transformer to effectively identify relevant features across various time scales, making it ideal for real-time applications. The architecture includes only the encoder part with a classifier's head to output probabilities of dog behavior. We used an open-access dataset focusing on seven different dog behavior, captured by motion sensors on top of the dog's back. Through experimentation and optimization, our model demonstrates superior performance with an impressive accuracy rate of 98.5%, outperforming time-series DNN models. The model's efficiency is further highlighted by its reduced computational complexity, lower latency, and smaller size, making it well-suited for deployment in resource-constrained environments.**

***Index Terms*—accelerometer, attention mechanism, dog behavior, dog activity detection, deep neural network, gyroscope, inertial sensors, pet activity detection (PAD), mode recognition, motion sensors, long short-term memory, machine learning, supervised learning, real-time, transformers.**

## I. Introduction

UNDERSTANDING dog behavior is crucial for their well-being and for creating positive relationships between dogs and humans [1]. Recognizing signs of fear, aggression, or anxiety can help prevent bites or other dangerous situations, highlighting the need for awareness of these behaviors for safety reasons [2]. Additionally, changes in a pet's behavior can indicate health problems, making early detection important for getting timely medical care [3], [4]. This knowledge is also key in dog training, where understanding the correlation between the dog's behavior can lead to more effective and humane training methods [5]–[7]. To avoid misunderstandings and ensure peaceful coexistence, it is important to consider the various ways dogs communicate through body language, sounds, and behavior [8]–[10].

The use of sensor technology and especially the use of motion sensors [1], [11]–[15] has become increasingly important in this field. Motion sensors offer a detailed way to observe and analyze dog behavior, enhancing the ability to monitor their health, emotional state, and interactions. The adoption of motion sensor technology in dog research and activity monitoring has seen significant applications, extending from domestic activity tracking to roles in search and rescue operations [16], [17]. Accelerometers and gyroscopes capture a dog's movements by measuring specific force and angular velocity, respectively. [12], [16], [18]. They are conveniently attachable to a dog's neck or back, offering a seamless method for gathering data on dog behavior in various environments [19].

A notable contribution to this field is the dataset released in [20], which encompasses motion sensor data from 45 dogs, each data sample tagged with specific behaviors. This rich dataset serves as a foundational resource for advancing our understanding of dog activities and the potential for behavioral prediction and monitoring.

Deep learning (DL) [21], a sophisticated branch of machine learning, simulates the neural circuitry of the human brain to process data and formulate patterns for decision-making. DL models are adept at managing complex, high-dimensional datasets with a level of efficiency that surpasses traditional machine learning methodologies. This paradigm shift has revolutionized multiple disciplines, including natural language processing (NLP), computer vision, and speech recognition, by offering models capable of autonomously learning feature representations directly from raw data, obviating the need for manual feature extraction.

Several studies explored dog behavior classification utilizing motion sensors using DL. In the seminal study by Kasnesis et al. [16], DL techniques were utilized on inertial sensors to facilitate real-time operations in dog rescue missions. The groundbreaking research by Hussain et al. [22] demonstrated the efficacy of a 1D convolutional neural network (CNN) [23], achieving an impressive accuracy of 96.85%. Notwithstanding, this model required a substantial number of parameters for its operation. Subsequent work [24] involved the deployment of a long short-term memory (LSTM) architecture [25], which garnered an accuracy of 94.25%. Further, in the study conducted by Eerdekens et al. [26], a novel methodology that utilized a reduced sampling rate of 10 [Hz] was introduced. Despite this lower rate, the study achieved an exceptional accuracy of 97.87%. Notably, this approach necessitated the inclusion of a considerable size of samples into the model, alongside a processing delay exceeding 10 seconds. This finding emphasizes the importance of model optimization to accommodate significant data loads and processing latencies.

The advent of transformer [27] has further propelled the capabilities of DL, particularly in natural language processing (NLP) and, of late, computer vision. Transformers are distinguished by their self-attention mechanisms, which enable dynamic allocation of importance to different segments of input data. This attribute renders them exceptionally suitable for processing sequential data, where discerning the context

Accepted to IEEE Sensors Journal: August 2024.
Barak Or, Haifa, 3499604, Israel (e-mail: Barakorr@gmail.com)

and relationships within the dataset is paramount. Additionally, transformers have demonstrated exceptional efficacy within the domain of time series analysis, a topic succinctly covered in [28].

The DL framework, especially when augmented with transformers, offers a refined understanding and modeling of sequential data and might have an advantage for identifying complex patterns of movement associated with various dog behaviors, as observed in other tasks involving sensor data [18], [29]–[35]. Inspired by the unparalleled efficacy of transformers across diverse domains, this research endeavors to leverage their potential to enhance the precision of behavior classification.

In our study, the methodological framework encompasses an analysis of the database provided by Vehkaoja et al. [20], followed by the training of a transformer-based classifier specifically tailored for our classification task involving motion data. The optimization process of the transformer is meticulously designed to enhance its performance in handling our dataset, with iterative refinements conducted until notable outcomes are achieved. Furthermore, a critical aspect of our methodology is the comprehensive evaluation of the model, taking into account the number of parameters, FLOPs (floating-point operations per second), model size, latency, and robustness consideration to ensure its suitability for real-time applications. The efficacy of our transformer-based approach is benchmarked against traditional models, including LSTMs, Bi-directional LSTM [36], and CNN-LSTM, highlighting the innovation and performance advantages of our work.

The main contributions of the paper are as follows:

1) Derivation of a framework for motion sensor classification with transformer. The proposed approach is the first to present a transformer-based architecture to serve as a general framework for general activity recognition in dog behavior classification.
2) A transformer network architecture, including the encoder part only, designed for handling motion sensor data with real-time considerations. Analysis of FLOPs, latency, and memory is provided.
3) Evaluation of the proposed framework on an open-access database and comparison with additional architectures to obtain state-of-the-art results.

This paper is structured as follows: Section II provides an overview of the transformer model and the attention mechanism. Section III discusses the learning methodology, covering data acquisition, database creation, loss function definition, training, and modern DNN models. Section IV focuses on the presentation of results and discussions, encompassing the definition of the error metric, a comparative analysis of the model's performance, and an examination of the influence of architectural design on real-time efficacy. Section V offers concluding remarks for the paper.

## II. Transformer-based Motion Sensor Classification Framework

This section reviews the transformer-based motion sensor classification framework, leveraging its remarkable success in NLP tasks to motion sensor real-time data classification task. We delineate the framework into 4 key components, forming the ensemble of our resultant DNN model. We begin with the attention mechanism, the core of the transformer architecture, enabling selective focus on relevant segments of the input sequence for enhanced feature discernment. Subsequently, we explore the multi-head attention (MHA) mechanism, facilitating simultaneous capture of diverse dependencies within the input sequence. Further, we discuss the encoder portion of the encoder-decoder transformer structure. Lastly, we address the absence of recurrence by reviewing the positional encoding, vital for preserving sequence order.

### A. Attention Mechanism

The attention mechanism serves as the cornerstone of the transformer architecture, enabling the model to selectively focus on relevant segments of the input sequence. This mechanism is pivotal in motion sensor data classification, as it allows the model to discern critical features that are indicative of different behavior modes exhibited by dogs.

At its core, the attention mechanism operates by computing a weighted sum of values, $\mathbf{V}$, based on the compatibility between a query $\mathbf{Q}$, and a set of keys, $\mathbf{K}$, from the input sequence. Mathematically, this is represented as:

$$\text{Attention}\left(\mathbf{Q},\mathbf{K},\mathbf{V}\right) = softmax\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}. \tag{1}$$

where $\mathbf{Q} \in \mathbb{R}^{N\times H\times S\times d_k}$, $\mathbf{K} \in \mathbb{R}^{N\times H\times S\times d_k}$, and $\mathbf{V} \in \mathbb{R}^{N\times H\times S\times d_v}$ are the query, key, and value tensors, respectively, for a batch of size $N$. Each tensor consists of $H$ attention heads, each with a sequence length of $S$, and a feature dimension of $d_k$ or $d_v$.

The softmax function is given by:

$$softmax(\mathbf{z}_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}, \tag{2}$$

where in our case is $\mathbf{z} = \frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}$. In the context of dog behavior classification, the query, key, and value vectors correspond to different aspects of the motion sensor data. The query represents a specific feature or aspect of the input sequence that the model is interested in, such as a particular pattern of motion indicative of a behavior mode. The keys encapsulate information from the entire input sequence, allowing the model to compare the relevance of different segments to the query. The values contain the actual sensor readings or features associated with each element of the input sequence.

By computing the attention scores between the query and keys, the model determines which parts of the input sequence are most relevant to the query. This enables the model to selectively attend to informative segments of the motion sensor data while downweighting irrelevant or noisy signals. For example, when classifying dog behavior modes, the attention mechanism might assign higher weights to sensor readings corresponding to rapid movements or changes in orientation, which are characteristic of specific behavior modes.

### B. Multi-Head Attention

The MHA mechanism divides the query, key, and value matrices into multiple heads, as illustrated in Figure 1, allowing the model to simultaneously attend to information from different representation subspaces at different positions (timestamps). This parallel processing capability of the MHA enables the model to capture a more comprehensive understanding of the input sequence, as given by:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, ..., \text{head}_H)\mathbf{W}^O \quad (3)$$

where each head is computed as:

$$\text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V) \quad (4)$$

and $\mathbf{W}_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $\mathbf{W}_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $\mathbf{W}_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $\mathbf{W}^O \in \mathbb{R}^{Hd_v \times d_{model}}$ are parameter matrices. Here, $d_{model}$ represents the dimensionality of the input feature vector.

The essence of MHA is to allow the model to focus on different parts of the input sequence. In the context of dog behavior classification through motion sensor data, this means the model can attend to the nuances of motion patterns, orientations, and other relevant signals that distinguish between different behaviors. For instance, one head might focus on the rhythm of a dog's gait, while another might prioritize the intensity of motion, enabling a more nuanced and comprehensive classification.

The key advantage of MHA is its ability to diversify the model's attention across different features and positions within the input sequence, which is critical for understanding complex, multimodal data.

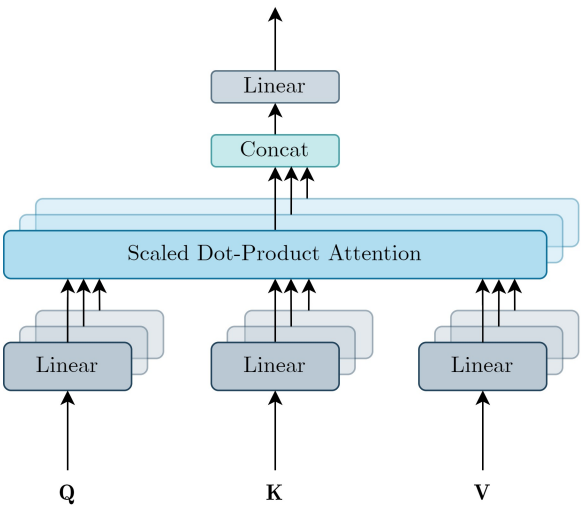


Fig. 1. Multi-Head Attention block: Central to transformer models. It segments the input into several heads, enabling parallel attention to diverse sequence segments, thereby enriching the model's interpretative capacity.

### C. Transformer Encoder

Our methodology uniquely harnesses the transformer encoder architecture, eschewing the decoder component. This strategic choice is predicated on the encoder's inherent capability to efficiently parse and encode sequential sensor data into a rich, feature-dense representation. The exclusion of the decoder is deliberate, aimed at refining the model's focus towards generating embedding that succinctly captures the essence of dog behaviors from motion sensor inputs without the need for sequence generation, which is the decoder's forte.

Central to our architecture is the transformer encoder, as presented in Figure 2. A sophisticated amalgam of layers, each comprising several key components: MHA, position-wise feed-forward networks (FFN), and residual connections, all normalized by layer normalization. The encoder processes input sequences—motion sensor data indicative of dog movements—by embedding them with positional information, preserving the temporal sequence's integrity (to be discussed in II.D).

The position-wise FFN within each encoder layer serves to apply non-linear transformations to the attention output, enhancing the model's ability to discern complex patterns and relationships in the data. Residual connections around each of the two main components (the MHA and FFN), followed by layer normalization [37], facilitate deeper model stacking by alleviating the vanishing gradient problem, thereby promoting stable and efficient training.

### D. Positional Encoding

In our classification task, positional encoding is crucial for embedding temporal information into the transformer' input, as the model inherently lacks sequential memory. This encoding enables the model to discern the order of motion sensor data, critical for identifying temporal patterns in dog behaviors.

The positional encoding formula is:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \quad (5)$$

where $pos$ is the sequence position, $i$ is the embedding dimension, and $d_{\text{model}}$ is the embedding size. This sinusoidal function embeds each position with a unique signal, allowing the model to capture sequence order and temporal dynamics.

## III. Learning Method

This section outlines our learning methodology for classifying dog behavior using motion sensors. It begins with the motivation for adopting a transformer-based architecture, followed by the dataset description - which details the motion sensor data from 45 dogs, focusing on seven distinct activities and the preprocessing steps taken. The loss function and the architecture are also described. Finally, the training optimization process is discussed.

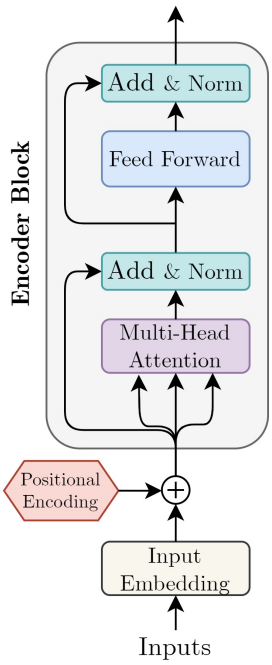


Fig. 2. Transformer encoder block illustration. The block consists of a multi-head attention mechanism followed by a feed-forward network, with residual connections and normalization applied at each stage.

### A. Motivation

Traditional ML and time series DNN-based models often face challenges in capturing temporal dependencies and efficiently processing high-dimensional sensor data. The transformer architecture, renowned for its self-attention mechanism, offers a potent solution by effectively identifying relevant features across various time scales and enabling parallel computations. This attribute is particularly beneficial for real-time applications, such as monitoring and interpreting dog behavior, where swift and accurate decision-making is very important.

### B. Dataset

The dataset, available from the National Library of Medicine, published by [20], comprises motion sensor data from 45 medium to large dogs, equipped with six-degree-of-freedom motion sensors on their collars and harnesses. Behavior annotations were added retrospectively, using video footage captured by two camcorders during experiments, with a granularity of one second. These annotations have been precisely aligned with the sensor data to provide a detailed and synchronized account of each dog's movements and behaviors.

A subset of this dataset was curated for analysis. Specifically, we focused on data sourced from the back sensor alone, encompassing seven distinct dog activities: walking, trotting, galloping, sniffing, lying on the chest, sitting, and standing. Each data instance comprises a 2-second segment of motion sensor signal along with a corresponding label indicating the dog's activity.

Our dataset underwent refinement to isolate activities directly indicative of dog behaviors. Activities labeled as 'Synchronization' and 'Extra Synchronization', which denote transitional or preparatory actions, were excluded from our analysis. Furthermore, the part of the data capturing instances of simultaneous activities was discarded from our research. Our focus remains on scenarios where a single activity is performed, typically represented in the first behavior. Additionally, rows containing undefined activities were removed from the dataset as they did not contribute meaningful information to our analysis. The data distribution is provided in Figure 3.

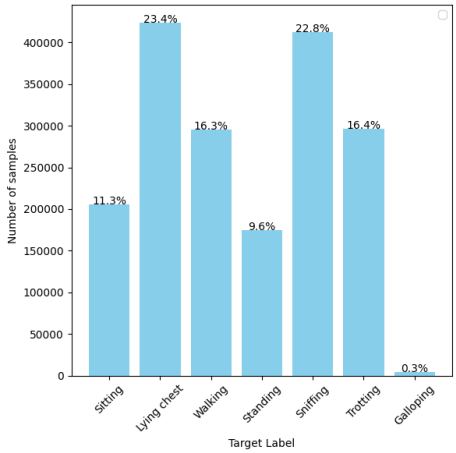


Fig. 3. The histogram representing the frequency distribution of seven distinct dog behavior classes.

Given that we are working with an imbalanced dataset, as the glopping class has a low amount of data, we ensured an equitable distribution of classes across the training, validation, and test sets through a technique known as stratified splitting [38]. First, we divided the entire dataset, denoted as $X_{\text{total}}$, into three subsets: $X_{\text{train}}$, $X_{\text{val}}$, and $X_{\text{test}}$. Each subset represented 64%, 16%, and 20% of the total data, respectively.

We identified unique Dog IDs in the dataset and made sure that each dog's data was properly represented in all subsets. For every Dog ID, its data was divided into three sets: training, validation, and test. The division was done in such a way that the distribution of classes remained consistent across all sets.

The training set, $X_{\text{train}} \in \mathbb{R}^{9,067\times200\times6}$, consisting of 9,067 examples, each comprising data in a $200 \times 6$ matrix format. Here, 200 timestamps are recorded, with 3 axes for accelerometer data and 3 for gyroscope data. Similarly, the validation set, $X_{\text{val}} \in \mathbb{R}^{2,265\times200\times6}$, comprises 2265 examples, each adhering to the same $200 \times 6$ structure. In total, we deal with 472 minutes of data in the entire dataset (302 minutes training set, 75.5 minutes validation set, and the rest 94.5 minutes for the test set).

In both datasets, the labels are represented as one-hot encoded vectors. For example, $\mathrm{y}_i = [0, 0, 0, 1, 0, 0, 0]$ represents class number 4.

### C. Loss Function

We are motivated by the work [39] and use the softmax cross entropy (CE) loss function for this multi-label classification task. In our context, where we have 7 classes, we input a one-hot encoding vector representing the ground truth labels. Given $y$ as the ground truth on-hot encoded vector of length 7 (the label) and $\hat{y}$ is the predicted probability distribution over the 7 classes, the binary cross-entropy loss, $\mathcal{L}$, for a single example can be expressed as:

$$\mathcal{L}(y, \hat{y}) = -\sum_{i=1}^{7} y_i \log(\hat{y}_i) \tag{6}$$

where the sum over $i$ ranges from 1 to 7, corresponding to the seven classes in our multi-label classification problem. For each class, i, the loss function calculates the cross-entropy between the predicted probability $\hat{y}_i$ and the actual label $y_i$. The term $y_i \log(\hat{y}_i)$ penalizes the model when it incorrectly predicts an outcome as being less likely (small $\hat{y}_i$).

### D. Architecture

The suggested architecture, as illustrated in Figure 4, comprises several layers in addition to the transformer encoder.

*1) Additional Layers Description:*

- **Feed-Forward Network (FFN):** Within each transformer block, the FFN is a fully connected neural network that applies a linear transformation followed by a non-linear activation function. It is defined as:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{7}$$

  where $W_1$, $W_2$, $b_1$, and $b_2$ are the weights and biases of the FFN, and the Rectified Linear Unit (ReLU) activation function, denoted as $\max(0, \cdot)$, applies a simple threshold operation to each input element, setting all negative values to zero while keeping all positive values unchanged.
- **Global Average Pooling (GAP):** In the context of time series data, GAP is used to condense the information across the entire sequence, by calculating the average value of each feature over all time steps.
- **Leaky ReLU:** Introduces non-linearity while allowing a small, non-zero gradient when the unit is not active. It is defined as:

$$\text{LeakyReLU}(\alpha) = \begin{cases} \alpha & \text{if } x > 0 \\ 0.01\alpha & \text{otherwise} \end{cases} \tag{8}$$

*2) Description of The Final Best-Performing Transformer-based DNN Model:* The input to the model consists of a signal captured at a sampling rate of 100 [Hz] over 2 seconds, resulting in input dimensions of 200 (time steps) by 6 (sensor channels). The architecture comprises 3 encoder blocks, each integrating an MHA mechanism and an FFN. The MHA is configured with 6 attention heads, each having dimensions of 32. Within each transformer block, the FFN is configured with dimensions of 1,024. The output from the final encoder block undergoes GAP to aggregate and condense the feature representation. Subsequently, this representation is passed through a linear layer consisting of 10 parameters. A Leaky ReLU activation function is then applied to introduce non-linearity and enhance the model's learning capability. The architecture concludes with a sigmoid activation function, which transforms the final output into a prediction vector with dimensions of 7 by 1, representing the probability distribution across the target classes or outcomes.

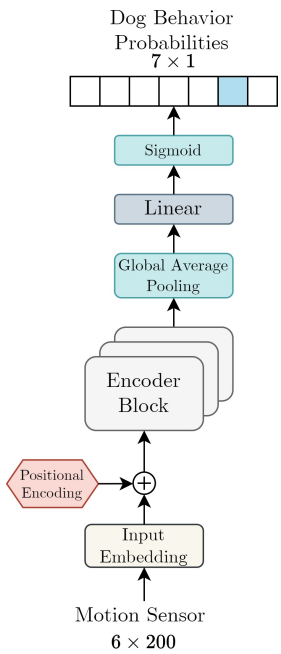


Fig. 4. DNN-based transformer architecture: Processes IMU signals (200x6) using three transformer encoder blocks, each with six attention heads (32-dimensional) and a 1024-dimensional feed-forward network. The architecture concludes with global average pooling, a linear layer (10 parameters), Leaky ReLU, and a sigmoid function, outputting a 7x1 prediction vector.

### E. Training

We iteratively train multiple models until reaching the final one. Each model is optimized using the ADAM optimizer [40] with an initial learning rate of 0.0002, trained for 1,000 epochs, with a batch size of 128. Additionally, we employ a learning rate schedule using an exponential decay function with decay steps of 300,000 and a decay rate of 0.05. All experiments were conducted on a Google Colab environment utilizing an A100 GPU.

## IV. Results and Discussion

This section presents a comprehensive analysis of the performance of the proposed transformer-based DNN model. It begins with a detailed explanation of error metrics such as precision, recall, F1-score, and accuracy, followed by a discussion on the results of the transformer model, emphasizing the impact of varying the number of transformer blocks on model performance. The section also includes a performance comparison with other models like LSTM, LSTM-CNN, and Bi-LSTM, highlighting the superior accuracy and computational efficiency of the transformer-based model.

### A. Error Metrics

Let TP, FP, TN, and FN represent the counts of true positives, false positives, true negatives, and false negatives, respectively. Precision is articulated as the ratio of TP to the sum of TP and FP, encapsulated by the equation:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{9}$$

highlighting the model's accuracy in identifying positive instances among all positively predicted cases.

Recall, or sensitivity, measures the proportion of actual positive cases correctly identified by the model, defined by:

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{10}$$

thereby assessing the model's ability to capture all relevant instances.

The F1-Score harmonizes the balance between precision and recall into a single metric, calculated as:

$$\text{F1-Score} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}, \tag{11}$$

offering a composite measure of the model's precision and recall capabilities.

Accuracy, delineating the overall correctness of the model, is delineated as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{12}$$

thus providing a comprehensive metric of performance across all classification categories.

### B. Transformer-based DNN Results

Transformer models have reached the forefront of performance, achieving an impressive accuracy rate of 98.5%. In our evaluation, we assessed how variations in the number of transformer blocks influence overall model efficacy. This analysis involved comparing 4 distinct configurations, each characterized by a different number of transformer blocks, ranging from 1 to 4.

Considering the real-time capability of the obtained model, a key metric in our examination was latency, defined as the interval between the input's introduction to the model and the delivery of a prediction. Additionally, we quantified the model's complexity by cataloging the total number of trained parameters, model size, and FLOPs, as indicators of each model's computational requirements. The results of all 4 models are summarized in Table 1. We found that the model equipped with 3 transformer blocks demonstrated superior performance, boasting the highest precision (94.7%), F1-Score (94.6%), and accuracy (98.5%). This model achieved these metrics with a latency of 74 [ms] and required computational resources amounting to 146,082 FLOPs, alongside 55,743 parameters.

In contrast, augmenting the model to include 4 transformer blocks did not proportionately enhance performance. While there was a marginal increase in recall (95.0%) and the F1-Score remained high (94.6%), the accuracy experienced a slight decline to 96.7%, with latency increasing to 84 [ms] and the computational demand soaring to 194,678 FLOPs, potentially exacerbating issues like overfitting.

Furthermore, the confusion matrix presented in Figure 5 illustrates a notable discrepancy in classifying galloping and trotting behaviors. The model predicted 'trotting' for 29% of instances labeled as 'galloping'. Despite this confusion in this smallest class (galloping), the accuracy for all other classes was close to 1, indicating robust performance across the spectrum of behaviors.

The FLOPs were calculated by considering both the forward pass and the backward pass during training, focusing on the operations within each transformer block—specifically, the matrix multiplications in the multi-head attention and feedforward layers.

The exemplary performance of the three-block transformer-based model, as well as the insights gleaned from the confusion matrix, highlight the importance of optimizing the number of Transformer blocks to balance computational efficiency with predictive accuracy.

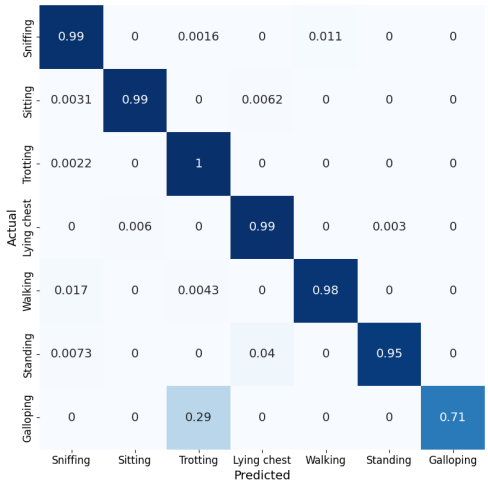


Fig. 5. Confusion matrix of the transformer-based model with 3 blocks.

### C. Model Performance Comparison

We compared the performance of our Transformer-based DNN model with other architectures, namely LSTM, LSTM-CNN, and Bi-LSTM, highlighting the unique strengths of our approach. The results are summarized in Table II. **LSTM:** This model employs a sequential LSTM architecture with layers of 32, 64, and 16 units, each followed by batch normalization and ending with LeakyReLU activation and global average pooling. The final layers consist of two dense layers with 300 and 10 units, respectively, each followed by LeakyReLU activation, and a final dense layer with 7 units.

**LSTM-CNN:** This architecture combines an LSTM layer with 32 units and four subsequent 1D convolutional layers with 128, 64, 64, and 64 filters, respectively. Each layer is followed by LeakyReLU activation, with a GAP layer. The network concludes with two dense layers with 300 and 10

TABLE I
TRANSFORMER-BASED DNN PERFORMANCE VS. NUMBER OF BLOCKS

| # Blocks | Precision | Recall | F1-Score | Accuracy | Latency [ms] | size [KB] | # Parameters | FLOPs |
|---|---|---|---|---|---|---|---|---|
| 1 | 80.3% | 78.1% | 75.2% | 80.1% | 76 | 73 | 18,679 | 48,890 |
| 2 | 91.5% | **98.9%** | 94.3% | 98.4% | 72 | 145 | 37,211 | 97,486 |
| 3 | **94.7%** | 94.6% | **94.6%** | **98.5%** | 74 | 218 | 55,743 | 146,082 |
| 4 | 94.2% | 95.0% | **94.6%** | 96.7% | 84 | 290 | 74,275 | 194,678 |

TABLE II
DNN MODEL PERFORMANCE

| Model Name | Precision | Recall | F1-Score | Accuracy | Latency [ms] | size [KB] | # Parameters | FLOPs |
|---|---|---|---|---|---|---|---|---|
| Transformer (ours) | 94.7% | 94.6% | 94.6% | **98.5%** | 74 | 218 | 55,743 | 146,082 |
| LSTM-CNN | 96.2% | 97.6% | 96.8% | 97.6% | 67 | 445 | 113,915 | 12,731,195 |
| LSTM | 96.4% | 97.8% | 97.0% | 97.6% | 68 | 170 | 43,579 | 2,728,379 |
| BiLSTM | **98.4%** | **97.9%** | **98.1%** | 97.8% | 73 | 42 | 10,967 | 384,801 |

units, respectively, each followed by LeakyReLU activation, and a final dense layer with 7 units.

**Bi-LSTM:** The Bi-LSTM model starts with a bidirectional LSTM layer with 32 units per direction, followed by batch normalization and LeakyReLU activation. A global average pooling layer is applied before the fully connected layers, which consist of a dense layer with 10 units followed by LeakyReLU activation, and a final dense layer with 7 units.

Our transformer-based DNN model achieves the highest accuracy of 98.5%, indicating its superior ability to correctly classify the input signals. While the Bi-LSTM model closely follows with an accuracy of 97.8%, the LSTM and LSTM-CNN models both have an accuracy of 97.6%. The Bi-LSTM model outperforms others in precision (98.4%) and recall (97.9%), suggesting its strong capability in identifying relevant instances and minimizing false positives. Although the transformer-based model has slightly lower precision and recall, its balanced performance ensures robustness across various scenarios. The Bi-LSTM model also leads in the F1-score (98.1%), indicating its overall effectiveness in classification. Our model, with an F1-score of 94.6%, still demonstrates commendable performance, considering its lower complexity and higher efficiency.

A notable advantage of our transformer-based model is its significantly lower number of parameters (55,743) and smaller size (218 KB), which make it highly suitable for deployment in resource-constrained environments. In contrast, the LSTM-CNN model has the highest number of parameters (113,915) and size (445 KB), as presented in Figure 6.

The FLOPs metric indicates the computational complexity of the models. Our transformer-based model has the lowest FLOPs (146,082), making it the most efficient model in terms of computational resources. The LSTM-CNN model, on the other hand, has the highest FLOPs (12,731,195), which could pose challenges in real-time processing and energy-constrained devices, as presented in Figure 6.

The transformer-based DNN model presents a compelling balance between high accuracy and computational efficiency. Its significantly lower complexity, reduced latency, and smaller model size make it an attractive choice for real-time applications and deployment on devices with limited computational resources.

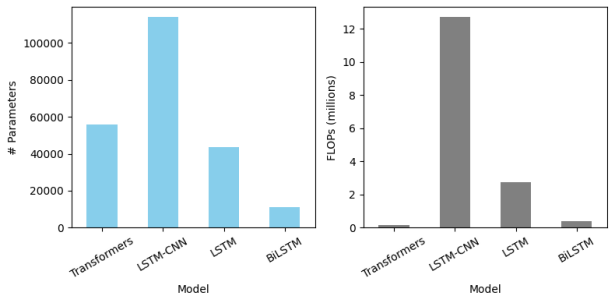


Fig. 6. Left: Histogram showing the number of parameters for each model. Right: FLOPs (in millions) per model.

### D. Robustness Tests

To evaluate model robustness, we conducted three tests: adding additive noise to the data, simulating missing data by replacing parts of the input with zeros, and retraining all four models using 10-fold cross-validation. These tests help assess the models' performance under various challenging conditions that can occur in a real-time environment.

**Additive Noise:** To thoroughly assess the robustness of the suggested transformer model, several tests were conducted to evaluate its ability to manage noisy data, specifically by adding Gaussian noise to the entire IMU signal across varying levels of variance. The motivation for this approach stems from real-world scenarios where sensor performance can degrade due to factors such as temperature fluctuations and other physical phenomena, particularly in low-cost MEMS sensors. Over time, these sensors may experience increased measurement noise, which can adversely affect their accuracy and reliability [18], [33], [35]. To simulate these conditions, we introduced Gaussian noise with a standard deviation ranging from 0.001 to 0.01 in each axis of the accelerometer and gyroscope signals. This emulation of noise degradation was applied not only to our proposed Transformer model but also to the other 3

models. The goal was to compare how well each model can withstand increasing noise levels that mimic the degradation of sensor data quality over time.

The figures below present a detailed comparison of the accuracy (Figure 7), precision (Figure 8), recall (Figure 9), and F1 score (Figure 10) for each model across different noise levels, highlighting the Transformer model's superior capability to handle noisy data.

The Transformer model shows significantly higher robustness to additive Gaussian noise compared to the other models. This suggests that the Transformer model could maintain better performance in real-life applications where sensor noise may increase due to environmental or operational factors.

**Missing Data:** The impact of missing sensor data on model performance was evaluated. We assessed how the unavailability of a portion of sensor information affects the models' ability to classify data accurately. To simulate missing data, we defined a proportion, $p$, for the $x$ channel of both the accelerometer and gyroscope. A given percentage of the data in each sensor was randomly replaced with zeros. For instance, if $p = 0.2$, then $20\%$ of the data in the $x$ channel for both the accelerometer and gyroscope would be replaced with zeros. We analyzed the effect of missing data for each channel separately and examined how the Transformer model, as well as the three other models, managed this challenge.

The plots below present a detailed comparison of the accuracy (Figure 11), precision (Figure 12), recall (Figure 13), and F1 score (Figure 14).

The results indicate that when $p$ is relatively small, $p < 0.2$, the Transformer model is generally able to maintain effective classification performance in most scenarios. While the Transformer model outperforms the LSTM and LSTM-CNN models in handling missing data, the BiLSTM model demonstrates a slight advantage over the Transformer in certain scenarios involving missing data in the x-axis for both the accelerometer and gyroscope.

**Cross Validation:** The entire training dataset was re-evaluated, excluding the designated test set, by performing a 10-fold cross-validation. This involves partitioning the training set into 10 equal-sized subsets (folds), where, in each iteration, 9 folds are used for training and the remaining fold is used for validation. This process is repeated 10 times. Table III presents the results, where the Transformer model obtained an accuracy of $88.3\% \pm 6.4\%$, followed by the highest accuracy of the BiLSTM, $88.7\% \pm 4.3\%$.

TABLE III
CROSS VALIDATION

| Model Name | Accuracy (mean $\pm$ std) % |
|---|---|
| Transformer (ours) | $88.3 \pm 6.4$ |
| LSTM-CNN | $80.3 \pm 1.4$ |
| LSTM | $80.12 \pm 3.4$ |
| BiLSTM | $\mathbf{88.7 \pm 4.3}$ |

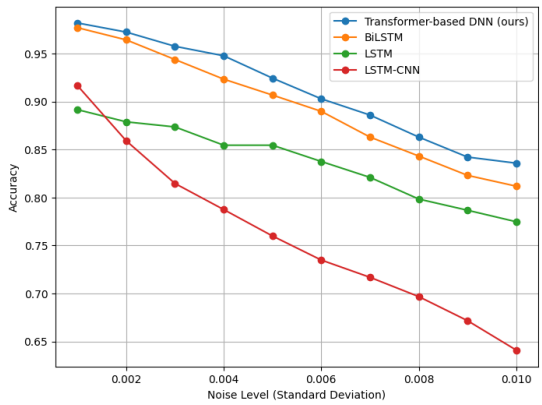


Fig. 7. Accuracy vs. noise level.

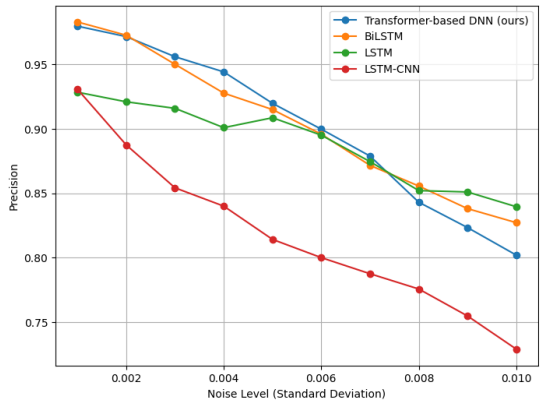


Fig. 8. Precision vs. noise level.

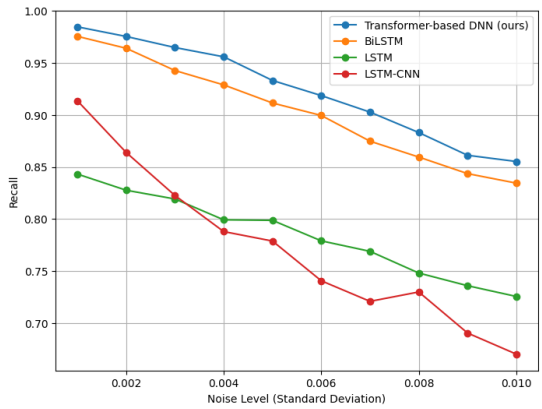


Fig. 9. Recall vs. noise level.

## V. CONCLUSIONS

Our study introduces a new approach to classifying dog behavior using motion sensor data through a transformer-based DNN model. The model's architecture, designed to capture temporal dependencies and efficiently process high-dimensional data, has demonstrated exceptional performance,

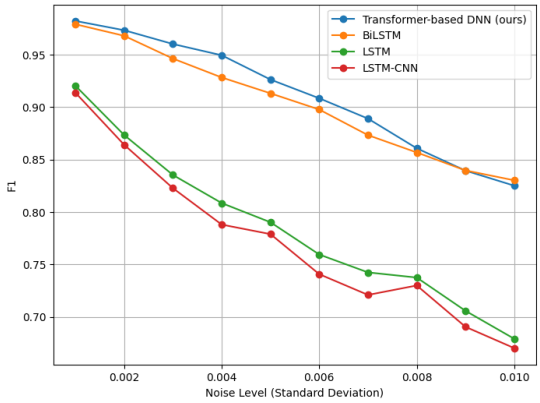


Fig. 10. F1 vs. noise level.

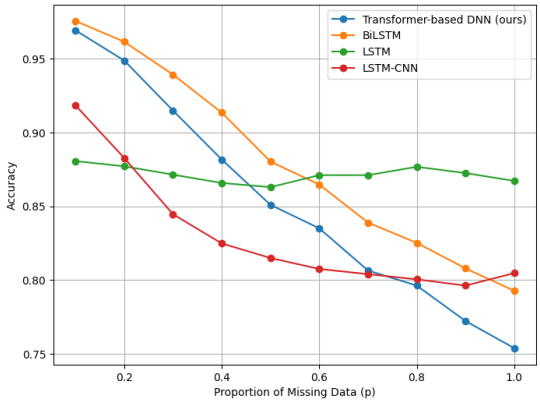


Fig. 11. Accuracy vs. missing data proportion.

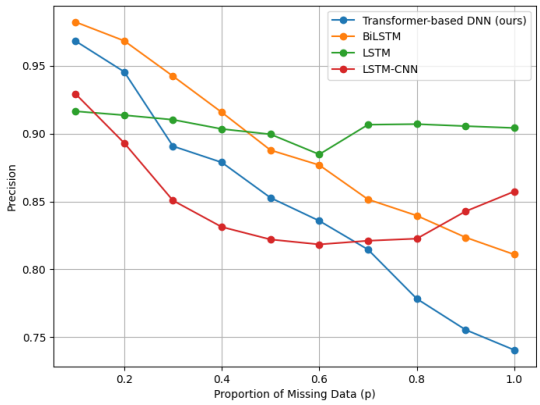


Fig. 12. Precision vs. missing data proportion.

achieving state-of-the-art accuracy of 98.5%. The comparative analysis with traditional models like LSTM, LSTM-CNN, and Bi-LSTM underscores the superiority of our approach, particularly in terms of computational efficiency and suitability for real-time applications. The model's reduced computational complexity and lower latency make it ideal for real-time applications. This innovation not only advances the field of motion

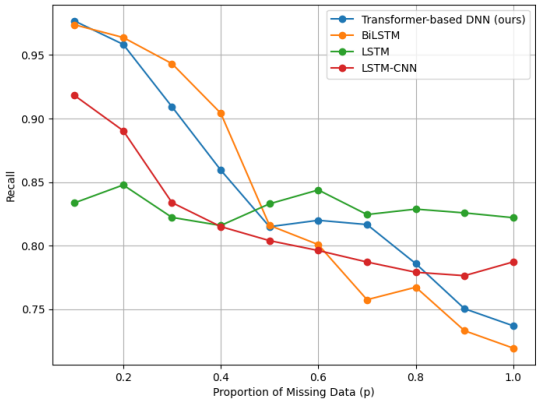


Fig. 13. Recall vs. missing data proportion.

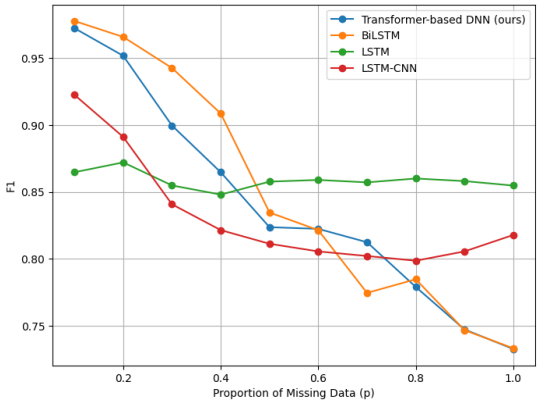


Fig. 14. F1 vs. missing data proportion.

sensor data processing but also holds significant potential for improving human-dog interactions by enabling early detection of behavioral issues and enhancing training methods.

The model's reduced computational complexity and lower latency make it ideal for real-time applications. Specifically, the model features a significantly lower number of parameters (55,743) and smaller size (218 KB), along with the lowest FLOPs (146,082), compared to traditional models. This translates to faster inference times, reduced training time, and lower memory requirements, making it suitable for deployment on resource-constrained devices such as mobile phones and embedded systems.

Our work paves the way for future research in the field of pet activity detection, offering a robust framework for real-time monitoring and analysis of dog behavior. The success of this study encourages the exploration of transformer-based models in other domains of motion sensor-based classification tasks.

## References

[1] L. Gerencsér, G. Vásárhelyi, M. Nagy, T. Vicsek, and A. Miklósi, "Identification of behaviour in freely moving dogs (canis familiaris) using inertial sensors," *PloS one*, vol. 8, no. 10, p. e77814, 2013.

[2] W. Boteju, H. Herath, M. Peiris, A. Wathsala, P. Samarasinghe, and L. Weerasinghe, "Deep learning based dog behavioural monitoring system," in *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*. IEEE, 2020, pp. 82–87.
[3] A. Hussain, S. Ali, M.-I. Joo, and H. C. Kim, "A deep learning approach for detecting and classifying cat activity to monitor and improve cat's well-being using accelerometer, gyroscope, and magnetometer," *IEEE Sensors Journal*, 2023.
[4] Y. Kamat and S. Nasnodkar, "Advances in technologies and methods for behavior, emotion, and health monitoring in pets," *Applied Research in Artificial Intelligence and Cloud Computing*, vol. 1, no. 1, pp. 38–57, 2018.
[5] H. Väätäjä, P. Majaranta, P. Isokoski, Y. Gizatdinova, M. V. Kujala, S. Somppi, A. Vehkaoja, O. Vainio, O. Juhlin, M. Ruohonen *et al.*, "Happy dogs and happy owners: Using dog activity monitoring technology in everyday life," in *Proceedings of the Fifth International Conference on Animal-Computer Interaction*, 2018, pp. 1–12.
[6] K. Ferres, T. Schloesser, and P. A. Gloor, "Predicting dog emotions based on posture analysis using deeplabcut," *Future Internet*, vol. 14, no. 4, p. 97, 2022.
[7] M. Foster, R. Brugarolas, K. Walker, S. Mealin, Z. Cleghern, S. Yuschak, J. C. Clark, D. Adin, J. Russenberger, M. Gruen *et al.*, "Preliminary evaluation of a wearable sensor system for heart rate assessment in guide dog puppies," *IEEE Sensors Journal*, vol. 20, no. 16, pp. 9449–9459, 2020.
[8] R. Deng, G. Zhou, L. Tang, C. Yang, and A. Chen, "E-docrnet: A multi-feature fusion network for dog bark identification," *Applied Acoustics*, vol. 220, p. 109950, 2024.
[9] J. Bradshaw, N. Rooney, and J. Serpell, "Dog social behavior and communication," *The domestic dog: Its evolution, behavior and interactions with people*, pp. 133–159, 2017.
[10] Y. S. Demirbas, H. Ozturk, B. Emre, M. Kockaya, T. Ozvardar, and A. Scott, "Adults' ability to interpret canine body language during a dog–child interaction," *Anthrozoös*, vol. 29, no. 4, pp. 581–596, 2016.
[11] E. Sazonov, *Wearable Sensors: Fundamentals, implementation and applications*. Academic Press, 2020.
[12] J. Farrell, *Aided navigation: GPS with high rate sensors*. McGraw-Hill, Inc., 2008.
[13] C. Ladha, N. Hammerla, E. Hughes, P. Olivier, and T. Ploetz, "Dog's life: wearable activity recognition for dogs," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, 2013, pp. 415–418.
[14] A. Muminov, M. Mukhiddinov, and J. Cho, "Enhanced classification of dog activities with quaternion-based fusion approach on high-dimensional raw data from wearable sensors," *Sensors*, vol. 22, no. 23, p. 9471, 2022.
[15] W. Chi-Pérez, J. Ríos-Martínez, F. Madera-Ramírez, and J. Estrada-López, "Wearable system for intelligent monitoring of assistance and rescue dogs," in *Journal of Physics: Conference Series*, vol. 2699, no. 1. IOP Publishing, 2024, p. 012001.
[16] P. Kasnesis, V. Doulgerakis, D. Uzunidis, D. G. Kogias, S. I. Funcia, M. B. González, C. Giannousis, and C. Z. Patrikakis, "Deep learning empowered wearable-based behavior recognition for search and rescue dogs," *Sensors*, vol. 22, no. 3, p. 993, 2022.
[17] Y. Wu, C. Nichols, M. Foster, D. Martin, J. Dieffenderfer, M. Enomoto, B. D. X. Lascelles, J. Russenberger, G. Brenninkmeyer, A. Bozkurt *et al.*, "An exploration of machine learning methods for gait analysis of potential guide dogs," in *The Tenth International Conference on Animal-Computer Interaction*, 2023, pp. 1–15.
[18] M. Freydin, N. Segol, N. Sfaradi, A. Eweida, and B. Or, "Deep learning for inertial sensor alignment," *IEEE Sensors Journal*, 2024.
[19] R. Amano and J. Ma, "Recognition and change point detection of dogs' activities of daily living using wearable devices," in *2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*. IEEE, 2021, pp. 693–699.
[20] A. Vehkaoja, S. Somppi, H. Törnqvist, A. V. Cardó, P. Kumpulainen, H. Väätäjä, P. Majaranta, V. Surakka, M. V. Kujala, and O. Vainio, "Description of movement sensor dataset for dog behavior classification," *Data in Brief*, vol. 40, p. 107822, 2022.
[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[22] A. Hussain, S. Ali, H.-C. Kim *et al.*, "Activity detection for the wellbeing of dogs using wearable sensors based on deep learning," *IEEE Access*, vol. 10, pp. 53 153–53 163, 2022.
[23] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.
[24] A. Hussain, K. Begum, T. P. T. Armand, M. A. I. Mozumder, S. Ali, H. C. Kim, and M.-I. Joo, "Long short-term memory (lstm)-based dog activity detection using accelerometer and gyroscope," *Applied Sciences*, vol. 12, no. 19, p. 9427, 2022.
[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[26] A. Eerdekens, A. Callaert, M. Deruyck, L. Martens, and W. Joseph, "Dog's behaviour classification based on wearable sensor accelerometer data," in *2022 5th Conference on Cloud and Internet of Things (CIoT)*. IEEE, 2022, pp. 226–231.
[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
[28] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," *arXiv preprint arXiv:2202.07125*, 2022.
[29] I. Dirgová Luptáková, M. Kubovčík, and J. Pospíchal, "Wearable sensor-based human activity recognition with transformer model," *Sensors*, vol. 22, no. 5, p. 1911, 2022.
[30] A. S. Kumar, S. Raja, N. Pritha, H. Raviraj, R. B. Lincy, and J. J. Rubia, "An adaptive transformer model for anomaly detection in wireless sensor networks in real-time," *Measurement: Sensors*, vol. 25, p. 100625, 2023.
[31] Z. Geng, Z. Chen, Q. Meng, and Y. Han, "Novel transformer based on gated convolutional neural network for dynamic soft sensor modeling of industrial processes," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1521–1529, 2021.
[32] M. A. Belay, A. Rasheed, and P. S. Rossi, "Mtad: Multi-objective transformer network for unsupervised multi-sensor anomaly detection," *IEEE Sensors Journal*, 2024.
[33] B. Or, "Carspeednet: A deep neural network-based car speed estimation from smartphone accelerometer," *arXiv preprint arXiv:2401.07468*, 2024.
[34] B. Or, N. Segol, A. Eweida, and M. Freydin, "Learning position from vehicle vibration using an inertial measurement unit," *IEEE Transactions on Intelligent Transportation Systems*, 2024.
[35] M. Freydin and B. Or, "Learning car speed using inertial sensors for dead reckoning navigation," *IEEE Sensors Letters*, vol. 6, no. 9, pp. 1–4, 2022.
[36] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural networks*, vol. 18, no. 5-6, pp. 602–610, 2005.
[37] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.
[38] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.
[39] A. Mao, E. Huang, H. Gan, R. S. Parkes, W. Xu, and K. Liu, "Cross-modality interaction network for equine activity recognition using imbalanced multi-modal data," *Sensors*, vol. 21, no. 17, p. 5818, 2021.
[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

**Barak Or** (Member, IEEE) received a B.Sc. degree in aerospace engineering (2016), a B.A. degree (cum laude) in economics and management (2016), and an M.Sc. degree in aerospace engineering (2018) from the Technion–Israel Institute of Technology. He graduated with a Ph.D. degree from the University of Haifa, Haifa (2022). His research interests include navigation, deep learning, sensor fusion, and estimation theory.