electronics

MDPI

Article

From Single to Deep Learning and Hybrid Ensemble Models for Recognition of Dog Motion States †

George Davoulos [1], Iro Lalakou [2] and Ioannis Hatzilygeroudis [1,*]

[1] Department of Computer Engineering & Informatics, University of Patras, 26504 Patras, Greece; georgedavoulos13@gmail.com
[2] Department of Informatics and Telecommunications, University of Peloponnese, 22100 Tripolis, Greece; irolalakou@gmail.com
* Correspondence: ihatz@ceid.upatras.gr
† Davoulos, G.; Lalakou, I.; Hatzilygeroudis, I. Recognition of Dog Motion States: Ensemble vs. Deep Learning Models. In Proceedings of the 15th International Conference on Information, Intelligence, Systems and Applications (IISA-24), Chania Crete, Greece, 17–19 July 2024.

Abstract: Dog activities recognition, especially dog motion status recognition, is an active research area. Although several machine learning and deep learning approaches have been used for dog motion states recognition, the use of ensemble learning methods is rather missing, as well as a comparison with deep learning ones. This paper focuses on the use of deep learning neural networks and ensemble classifiers in recognizing dog motion states and their comparison. A dataset from the Kaggle database, which includes measures by accelerometer and gyroscope and concerns seven dog motion states (galloping, sitting, standing, trotting, walking, lying on chest, and sniffing), was used for our experiments. Gaussian Naive Bayes, Decision Tree, k-Nearest Neighbors (kNN), Random Forest, a Bagging Tree-Based Classifier, a Stacking Classifier, a Compound Stacking Model (CSM), a Convolutional Neural Network (CNN), a Recurrent Neural Network (RNN), and a Hybrid Cascading Model (HCM) were used in our experiments. Results showed a 1.78% superiority in accuracy (92.64% vs. 90.86%) of deep learning (RNN) vs. stacking (CSTAM) best classifier, but at the cost of larger complexity and training time for the deep learning classifier, which makes ensemble techniques still attractive. Finally, HCM gave the best result (96.82% accuracy).

Keywords: dog motion states recognition; ensemble learning; deep learning; stacking model; RNN; cascading model

check for updates

Academic Editor: Yue Wu

Received: 3 April 2025
Revised: 6 May 2025
Accepted: 8 May 2025
Published: 9 May 2025

Citation: Davoulos, G.; Lalakou, I.; Hatzilygeroudis, I. From Single to Deep Learning and Hybrid Ensemble Models for Recognition of Dog Motion States. *Electronics* **2025**, *14*, 1924. https://doi.org/10.3390/electronics14101924

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The recognition of animal motion using technical means represents an evolving field of research [1]. The continuous progress in artificial intelligence (AI) and machine learning (ML) has facilitated the development of advanced algorithms that detect and recognize animal motion status with accuracy and reliability. By using data from sensors such as cameras, accelerometers, and gyroscopes, those algorithms analyze motion patterns of animals, providing significant insight into their behavior, environmental changes, and interaction with nature [2]. The use of machine learning facilitates data analysis and the identification of important information in animal behavior. The ability to recognize specific behaviors, such as playful behavior or signs of health problems, is achievable through building suitable models by machine learning techniques [3].

The results of this kind of research are useful in various application domains. In ecology, animal motion recognition can aid in the study of movements, feeding behavior, and

reproductive behavior of species [4]. Moreover, in wildlife conservation, motion detection can help locate and protect endangered species [5]. Finally, the application of motion recognition can also be used in geographical monitoring, food security, and even entertainment by monitoring animal behavior in zoos and aquariums [6]. This is a critical task given the diverse roles dogs play in various human activities, including search and rescue, assistance, and therapy [7].

The focus of this paper is to address the specific problem of recognizing dog motion states, based on measures from the accelerometer and gyroscope. Although several ML methods, such as SVM and kNN [8] (which could be called classical methods), and deep learning (DL) techniques, such as Convolutional Neural Networks (CNN) [9] and Long Short-Term Memory (LSTM) Networks [10], have been employed for animal motion recognition, there exists a notable gap in the literature concerning use of ensemble methods (Boosting, Bagging, Stacking) or other compound models for animal, and especially dog, motion states recognition. It is evident that ensembles have achieved better results than single classifiers in other application domains, e.g., disease recognition [11], sentiment analysis [12], etc. So, the use of ensembles for dog motion status recognition and comparison with deep learning methods sounds quite interesting.

The primary objectives of this study are to identify and predict dog motion states utilizing a range of machine learning algorithms (from simple/single to more complicated) and to conduct a comparative analysis of those classifiers, as well as a comparison of the winner with the state-of-the-art models. Through the rigorous evaluation and comparison of those algorithms, this research aims to identify the most appropriate approaches for predicting dog motion states, thereby contributing to the field of motion state recognition and enhancing the application of machine learning techniques to address practical challenges in animal behavior analysis.

The algorithms which are examined are: Gaussian Naive Bayes, Decision Tree, k-Nearest Neighbors (kNN), Gradient Boosting, Random Forest, a Bagging Tree-Based Classifier, two Stacking Classifiers, a Compound Stacking Model, a Convolutional Neural Network (CNN), a Recurrent Neural Networks (RNN), and a Hybrid Cascading Model.

This paper is an extension of [13] in the following directions:

- An extended dataset that concerns seven dog motion states, rather than five as in [13], is used.
- A new stacking model, called the Compound Stacking Model (CSM), is configured and used.
- A new hybrid cascading model (HCM), combining RNN and CSM models, is introduced and used. HCM produces the best accuracy result that overcomes almost all SOTA systems.
- Comparison with old and new state-of-the-art models is presented.
- Useful conclusions about the scalability of the used models and the recognition difficulty level of dog motion states are drawn.

The structure of the paper is as follows: Section 2 outlines the related work in the field. Section 3 describes the dataset and the preprocessing tasks, as well as the experimentation methodology. Section 4 presents the experimental results, while Section 5 includes comparisons and discussion on them. Finally, Section 6 concludes the paper.

2. Related Work

Recent research in dog motion recognition, leveraging wearable sensors and deep learning, has made significant strides.

In [14], the authors use their own dataset, which was produced from recordings of activities of 10 dogs of different genders, breeds, ages, and sizes via accelerometer and

gyroscope put on the neck and the tail of the dogs. Seven dog states/activities (walking, sitting, stay, eating, sideway, nosework, jump) are of concern. A class weight technique is used for balancing data. Five ML algorithms were applied: Random Forest, SVM, kNN, Naive Bayes, and ANN. ANN achieved the best accuracy (96.58%) and F1-score (93.65%). No configuration information about the ANN is provided.

The study in [15] analyzes a dataset of over 50,000 frames, labeling crowd and eating/drinking behaviors of dogs. Machine learning techniques, specifically Convolutional Neural Network (CNN) and Support Vector Machine (SVM), were employed to classify behaviors such as drinking, eating, licking an object, licking itself, petting, rubbing, scratching, shaking, and sniffing. The best results achieved were for SVM, with an accuracy of 97.2% for the crowd dataset and an overall accuracy of 91.9% for the eating/drinking dataset.

Authors of [16] first present the process of creation of a dataset for dog motion states recognition using measurements of an accelerometer and gyroscope attached to the neck and the back of 45 dogs of various breeds, regarding seven states (galloping, sitting, standing, trotting, walking, lying on chest, and sniffing), the same as those used in our work. The dataset has been uploaded to the Mendeley database and is described in detail in [17]. It presents the result of using four machine learning methods (LDA, QDA, SVM, Tree) for predicting dog motion states. The best result was achieved by the SVM classifier on data from both the accelerometer and the gyroscope, namely an accuracy of 91.4%. As noticed, Sniffing was the best recognized motion state.

The system in [18] recognizes six dog states/activities: standing, walking, running, sitting, lying, and resting. Two datasets were created, regarding data about activities/states of 18 domestic dogs of various breeds, ages, and weights. One included data from the accelerometer and gyroscope (on a smart costume on the body of the dog), and the other from (calculated) quaternion values of dog movements. Four ML techniques are used: SVM, kNN, DT, and GNB (Gaussian Naive Bayes). GNB achieved the best average F-scores for both datasets, 88% for the first and 93% for the second.

The authors in [19] use their own dataset, which was produced from recordings of activities of nine dogs of different breeds, ages, and sizes (all females except one) via accelerometer and gyroscope put on the neck (collar) and the back of the dogs. It refers to five dog states/activities (walking, sitting, standing, lying down, running). A 1D CNN architecture is used, consisting of two convolutional layers, two max pooling layers, two dropout layers, one flatten layer, and three fully connected layers. The average accuracy achieved is 92.60%.

Also, in [9], the authors use their own dataset, which was produced from recordings of activities of 10 dogs of different genders, breeds, ages, and sizes via accelerometer and gyroscope put on the neck and the tail of the dogs. It deals with ten dog states/activities (walking, sitting, down, staying, eating, sideway jumping, running, shaking, nosework). A class weight technique is used for balancing data. A 1D CNN architecture is used, consisting of five convolutional layers, two dropout layers, one flatten layer, and three fully connected layers. The achieved average accuracy is 96.85%, and the F1-score is 97% (not reported in the paper-calculated by us from the class metrics).

An LSTM deep learning model is used in [10], consisting of six LSTM layers, three dropout layers, and three fully connected layers with a Softmax activation function at the output layer. Adam optimizer was used in training. It seems that the same dataset as in [7] is used here. A class weight technique is used for balancing data. An accuracy of 94.25% was achieved.

In [20], the study uses a dataset collected from a 4-year-old male Yorkshire terrier, consisting of natural behaviors observed within a 30-min video, and sensor data collected from a wearable device. There are seven included behaviors: standing, sitting, lying with

raised head, lying without raised head, sniffing, walking, and running. Three machine learning algorithms, FasterRCNN, YOLOv3, and YOLOv4, were employed for dog detection and behavior recognition using video data. YOLOv4 achieved the highest detection rate at 72.01%. For sensor data, a combination of five statistical features (mean, variance, standard deviation, amplitude, and skewness) was used to improve performance. The best-performing model was a CNN-LSTM hybrid model, which achieved an accuracy of 93.4% when using YOLOv4 for dog detection.

The study in [21] utilizes an experimental dataset consisting of accelerometer data at a sampling rate of 10 Hz from six different dogs performing eight distinct activities (lying, sitting, standing, walking, running, sprinting, eating, drinking). Three machine learning models, namely a Random Forest classifier, a Convolutional Neural Network (CNN), and a hybrid CNN, were employed to classify the dog's activities. The authors experimented with different sampling rates of the dataset. The hybrid CNN achieved the best performance, with an overall classification accuracy of 96.90%.

The paper in [22] presents a method for predicting pet behaviors using a Time-Noise Generative Adversarial Network (TN-GAN) and a CNN-LSTM hybrid model. The dataset consisted of sensor data collected from 10 pets using wearable devices equipped with nine-axis sensors, with a total of 26,912 data points collected at a frequency of 50 Hz. Nine behavioral states were recorded: standing on two legs, standing on four legs, sitting on two legs, sitting on four legs, lying on the stomach, lying on the back, walking, sniffing, and eating. Utilizing this dataset, the augmented nine-axis sensor data demonstrated a high accuracy of 97% in behavioral prediction.

The paper in [23] also presents the process of the creation of a dataset for dog motion states recognition, as [15] does, using measurements of accelerometer, gyroscope, and magnetometer placed on the neck, back, and chest of 42 dogs of various breeds. Finally, the magnetometer features were removed from the dataset, given that initial experiments showed that they were not predictive enough. The dataset regarded five states (three static: Standing, Sitting, Lying down, and two dynamic: Walking, Body shake). Three different cascade classifiers, based on Random Forest, were used for predicting dog motion state. The best result was achieved by the third classifier, namely an F1-score of 90%. As noticed, Sniffing was the best recognized motion state.

Finally, the authors of [24] present a transformer-based DNN for dog motion state recognition. DNN consists of three encoder blocks, a ReLU-based fully connected feed-forward network (FFN) within each transformer block, a global average pooling (GAP) level, a linear level with Leaky ReLU, and a Sigmoid level. A subset of the Mendeley dataset (introduced in [15]) is used concerning the same dog motion states as in our system, but with quite a different number of samples per state. Adam optimizer is used during training, and finally, the DNN achieves an accuracy of 98.5% and an F1-score of 94.6%. Despite this, the most difficult to predict was galloping (accuracy 71%).

3. Materials and Methods

3.1. Datasets and Preprocessing

The dataset used in this research consists of measurements from wearable motion sensors, namely accelerometer and gyroscope, aimed at classifying typical dog activities in a semi-controlled test environment. This dataset is part of the "Dog Behavior Analysis Dataset" from the Kaggle database (https://www.kaggle.com/datasets/arashnic/animal-behavior-analysis (accessed on 9 February 2025)).

According to the data information sheet, data collection of the dataset involved forty-five medium- to large-sized dogs. Each dog was equipped with two sensor devices, one attached to the back in a harness and the other to the neck collar. Testing occurred in

a 10 m × 18 m dog gym arena, covered with artificial turf. The test sequence included seven tasks, where owners guided their dogs as directed. These comprised three static tests (such as sitting, standing, lying) and four dynamic tests (including slow walking, walking, playing, and treat searching), each lasting three minutes. After a short break, the sequence was repeated with a changed order of tasks. Dogs alternated between static and dynamic activities, concluding with a treat search involving sniffing out scattered dry dog food pieces. Dogs were fitted with two ActiGraph GT9X Link activity sensors, which included 3-axis accelerometers and gyroscopes (sampling rate: 100 Hz). One sensor was placed in a neoprene pocket on the back harness, and the other was securely attached to the collar's underside. This dataset seems to be very similar to that uploaded in Mendeley database (https://data.mendeley.com/datasets/vxhx934tbn/1 (accessed on 9 February 2025)), given that the descriptions of their creation processes are almost identical.

Characteristic images of the dog motion states dealt with in our dataset are depicted in Figure 1. Definitions of those motion states are presented in Table 1, taken from the Kaggle database's "Behavior Description" section.



Figure 1. Characteristic examples of dog motion states.

Table 1. Description of dog motion states.

Motion State	Description
Galloping	A 3- or 4-beat gait where the dog lifts and puts down both front and rear extremities in a coordinated manner, in 1-2-3-beat gait (canter) or in 1-2-3-4 beat gait (gallop). All four extremities are simultaneously in the air at some point in every stride. Galloping occurred only during the Playing task.
Sitting	The dog has four extremities and rump on the ground. The dog can change the balance point from central to hip or vice versa.
Standing	The dog has the four extremities on the ground, without the dog's torso touching the ground.
Trotting	A 2-beat gait where the dog lifts and puts down extremities in diagonal pairs at a speed faster than walking.
Walking	A 4-beat gait where the dog moves extremities at slow speed, legs are moved one by one in the order: left hind leg, left front leg, right hind leg, and right front leg. The dog moves straight forward or at a maximum angle of 45 degrees.

Table 1. *Cont.*

Motion State	Description
Lying on chest	The dog's torso is touching the ground, and its hips are at the same level as its shoulders. The dog can change balance point without using its limbs.
Sniffing	The dog has its head below its back line and moves its muzzle close to the ground. The dog walks, stands, or performs another slow movement, but its chest and bottom do not touch the ground. Taking food from the ground and eating it can be included (eating was not coded separately).

Dataset preprocessing included removing data that does not provide any information and selecting the most relevant motion states from the rest behaviors to ensure adequate data preparation before training the models. To test the reliability and validity of our proposed classifiers, we created two datasets, one with five motion states (Galloping, Sitting, Standing, Trotting, Walking), called the dataset-5 ms, and the other with seven motion states (the above, plus Lying on chest and Sniffing), called the dataset-7 ms, out of the seventeen dog behaviors recorded in the initial dataset.

The initial dataset included columns, as depicted in Table 2.

Table 2. Original dataset columns.

Column	Description
Dog ID	Number of dog ID
Test Num	Number of the test {1, 2}
t_sec	Time from the start of the test (in sec)
ABack_x	Accelerometer measurement from the sensor in the back, x-axis
ABack_y	Accelerometer measurement from the sensor in the back, y-axis
ABack_z	Accelerometer measurement from the sensor in the back, z-axis
ANeck_x	Accelerometer measurement from the sensor in the neck, x-axis
ANeck_y	Accelerometer measurement from the sensor in the neck, y-axis
ANeck_z	Accelerometer measurement from the sensor in the neck, z-axis
GBack_x	Gyroscope measurement from the sensor in the back, x-axis
GBack_y	Gyroscope measurement from the sensor in the back, y-axis
GBack_z	Gyroscope measurement from the sensor in the back, z-axis
GNeck_x	Gyroscope measurement from the sensor in the neck, x-axis
GNeck_y	Gyroscope measurement from the sensor in the neck, y-axis
GNeck_z	Gyroscope measurement from the sensor in the neck, z-axis
task	The task given at the time, <undefined> when no task is being performed
behavior_1	Annotated behavior 1, maximum of three simultaneous annotations at the same time
behavior_2	Annotated behavior 2, maximum of three simultaneous annotations at the same time
behavior_3	Annotated behavior 3, maximum of three simultaneous annotations at the same time
PointEvent	Short events annotated separately (Bark, for example)

Some columns were removed as they were irrelevant to motion recognition or performance comparison. Columns omitted include DogID, TestNum, t_sec, task, and PointEvent. The dataset was then restructured as depicted in Figure 2.

Behavior recording columns behavior_1, behavior_2, and behavior_3 could record one of various behaviors per data row. At first, we selected only five motion-related behaviors, including Walking, Trotting, Standing, Sitting, and Galloping. A new column, 'Behavior', was created to store these motion states, identified from the original behavior columns, replacing behavior_1, behavior_2, and behavior_3 (see Figure 3). Any row without motion data in the 'Behavior' column was subsequently removed. The size of the final dataset was 3,308,476 rows, distributed among the five motion states, as depicted in Figure 4. This is the

dataset-5 ms, which we dealt with in [13], where we trained single ML and DL models, as well as ML ensemble models based on various machine learning algorithms, and evaluated their performance.

	ABack_x	ABack_y	ABack_z	ANeck_x	ANeck_y	ANeck_z	\
7000000	-0.104004	0.043457	1.014160	-0.218262	0.608887	-0.625000	
7000001	-0.094238	0.048828	1.027344	-0.230957	0.639648	-0.651367	
7000002	-0.092285	0.055664	1.016602	-0.263184	0.742676	-0.732422	
7000003	-0.052734	0.125977	1.050293	-0.274902	0.764160	-0.763672	
7000004	-0.078613	0.146484	1.027832	-0.271973	0.710938	-0.783203	

	GBack_x	GBack_y	GBack_z	GNeck_x	GNeck_y	GNeck_z	\
7000000	37.231448	22.216798	-8.544922	-58.898929	-34.118654	27.954103	
7000001	31.921389	20.507814	-7.873536	-49.194339	-33.020022	22.460939	
7000002	32.348635	16.540528	-6.469727	-37.292483	-27.343752	13.671876	
7000003	33.630373	15.197755	-5.432129	-31.066896	-20.507814	4.211426	
7000004	30.517580	23.437501	-6.469727	-28.869631	-21.362306	-0.366211	

	Behavior_1	Behavior_2	Behavior_3
7000000	<undefined>	<undefined>	<undefined>
7000001	<undefined>	<undefined>	<undefined>
7000002	<undefined>	<undefined>	<undefined>
7000003	<undefined>	<undefined>	<undefined>
7000004	<undefined>	<undefined>	<undefined>

Figure 2. Dataset structure after redundant column removal.

	ABack_x	ABack_y	ABack_z	ANeck_x	ANeck_y	ANeck_z	GBack_x	GBack_y	GBack_z	GNeck_x	GNeck_y	GNeck_z	Behavior
6005161	0.388184	0.600098	1.202637	-0.312988	-0.008789	-0.795410	-40.222170	181.762706	108.337409	46.997073	59.509281	-57.495121	Walking
6005162	0.428223	0.591309	0.979980	-0.301758	0.020508	-0.739746	-52.246097	203.002942	98.632819	63.903812	44.616702	-59.265140	Walking
6005163	0.253418	0.501953	0.784668	-0.309082	0.060547	-0.723145	-83.007818	180.664074	83.190923	77.575688	32.653811	-58.166507	Walking
6005164	-0.052734	0.477051	1.046875	-0.337891	0.088867	-0.731445	-105.590827	171.142589	65.307621	87.280279	31.555178	-54.321292	Walking
6005165	-0.288574	0.581543	1.354980	-0.355469	0.095215	-0.753906	-115.173347	190.795910	37.963870	96.191412	39.001467	-46.813968	Walking
...	...	...	...	...	...	...	...	...	...	...	...	...	...
6990555	-0.147949	-0.058594	0.767090	-0.634277	-0.126465	-0.251953	-65.673832	63.720707	242.248550	-94.787603	40.100100	-80.810552	Walking
6990556	-0.206055	0.013184	0.830078	-0.563965	-0.238770	-0.224121	-53.710941	41.503909	261.047379	-93.505865	29.846193	-67.993168	Walking
6990557	-0.209961	0.056641	0.778809	-0.525391	-0.363770	-0.240723	-28.991701	-5.676270	272.094743	-81.604009	19.897462	-59.326175	Walking
6990558	-0.168457	0.073242	0.765137	-0.508789	-0.422852	-0.299316	0.793457	-55.114749	275.817888	-55.725101	13.488770	-52.185062	Walking
6990559	-0.146484	0.026855	0.854492	-0.472168	-0.416992	-0.380859	27.343752	-83.740239	282.165545	-25.817873	14.709474	-46.264651	Walking

Figure 3. Final dataset structure.

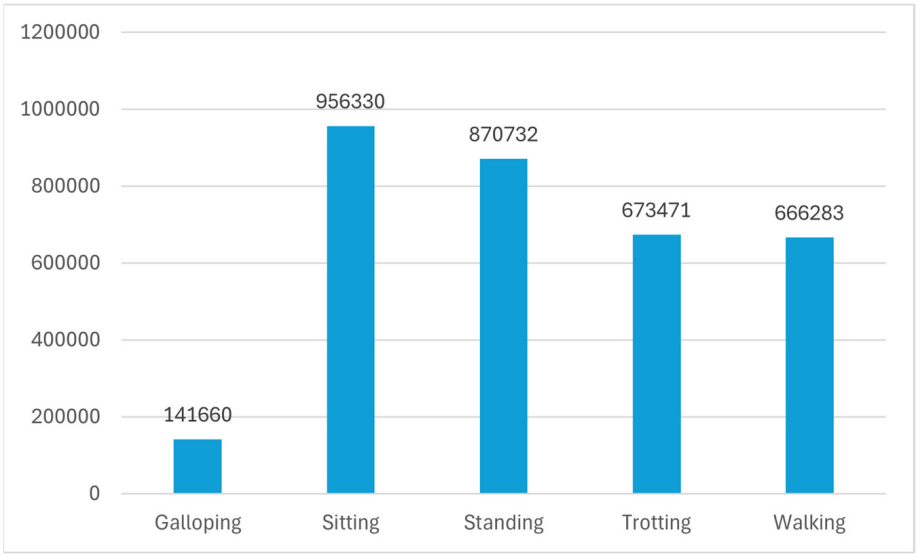


Figure 4. Distribution of dog motion states in dataset-5 ms. (y-axis: number of instances in the dataset).

Afterwards, we expanded dataset-5 ms by including two more motion states, Lying on chest and Sniffing. Thus, we created, in the same way, *dataset-7 ms*. The resulting dataset has 6,035,746 rows, and its structure is depicted in Figure 5. This is the dataset we are dealing with in this paper. The objective is to test the reliability and validity of the classifiers, which were configured and tested on dataset-5 ms and dataset-7 ms, and design new ones.

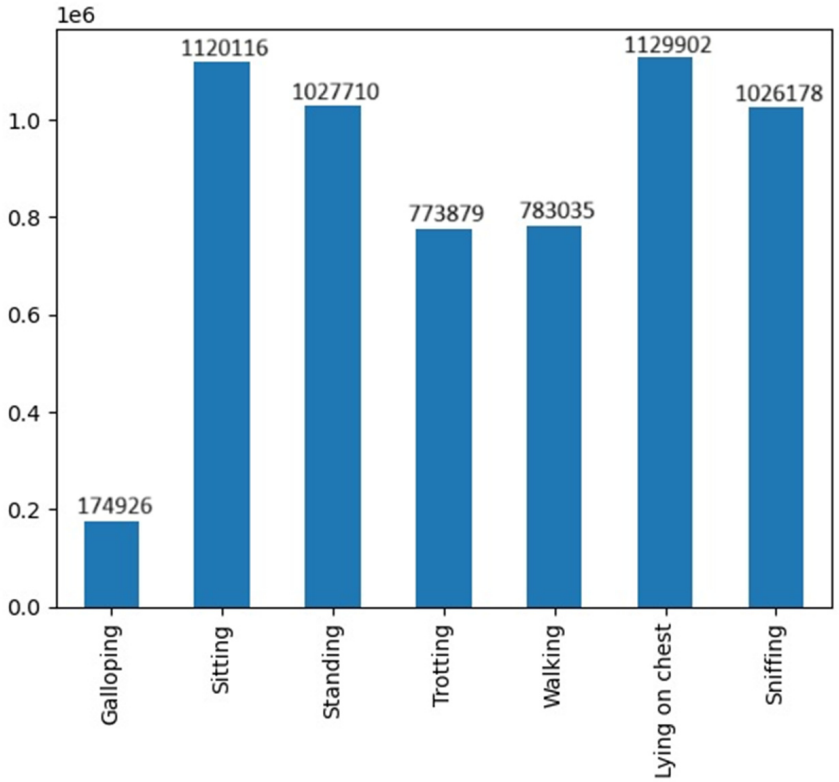


Figure 5. Distribution of dog motion states in dataset-7 ms. (y-axis: number of instances in the dataset).

3.2. Experimental Methodology

The general methodological approach followed in our experiments is as follows:

1. Application of the same classification models as those configured and tested for datset-5 ms in [13] to dataset-7 ms, so that we can see their behavior at an extended dataset.
2. Analysis of the new results and comparison with the old ones.
3. In case of non-satisfactory results, design and train new compound models.

In implementing the above approach, we first applied the three single classifiers of different nature to dataset-7 ms: Gaussian Naive Bayes (GNB), a Decision Tree (DT) algorithm (based on the CART algorithm), and the k-Nearest Neighbors (k-NN) algorithm.

Afterward, we applied the following ensemble models (the same as those in [13]):

- Random Forest (RF).
- Bagging model (BM) with DT as base classifier.
- Stacking model (SM) with k-NN and DT as base models, and LR as meta-classifier.

Next, we applied two selected DL architectures of a different nature:

- Convolutional Neural Network (CNN)
- Recurrent Neural Network (RNN)

The results were not better than the corresponding ones in [13]. They were 2% to 4% worse. So, we decided to configure a new stacking model, as follows:

- Use an ensemble model as an extra base model in SM, thus creating a compound stacking model (CSM). As a compound stacking model, we define the one that includes one (or more) stacking model(s) as base classifier(s) or meta-classifier. We chose Gradient Boosting (GB) as an extra base model, because this model had the best results from all ensemble models in [13].

Given that the results of CSM were still not satisfactory, we tried a new option:

- Combine the best ensemble model with the best deep learning model in a cascading mode, thus creating a hybrid cascading model (HCM). As a hybrid cascading model, we define a cascading model that mixes (conventional) ML models with DL models. This gave us the best result.

The above methodology is depicted in Figure 6, where R_{SIi}, R_{ENi}, and R_{DLi} represent the results of single, ensemble, and deep learning classifiers, respectively, with R_{BEST} being the best of them.

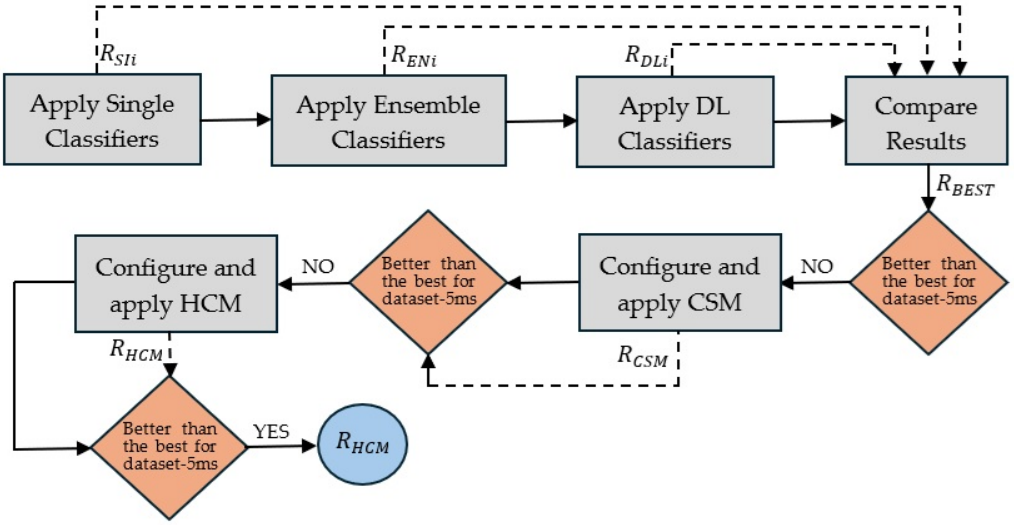


Figure 6. Experimental methodology for dataset-7 ms.

3.3. Implementation Tools and Metrics

In this study, the Jupyter Notebook 1.1.1. environment, integrated within the Anaconda platform and operating on Windows, was utilized for the development and evaluation of various classification methods. This environment enables Python code development and execution, coupled with the ability to display results, graphs, and comments, thus providing a comprehensive development setting for methodological analysis. Implementation was facilitated by corresponding Python libraries, ensuring reliable and efficient algorithm execution.

To evaluate the above machine learning models, we have used well-known metrics: precision, recall, F1-score, and accuracy [25].

4. Experimental Results

In this section, we present the experimental results of the application of various classifiers to the dataset-7 ms with reference to their results from the dataset-5 ms case [13]. The confusion matrices of the experiments are presented in Appendix A in the form of heatmaps.

Naive Bayes Model. The Gaussian Naive Bayes (GNB) classifier, employed with default settings, as in the dataset-5 ms case, achieved an average accuracy of 74% in motion states recognition.

In contrast to the 5 ms-datset case, the Sniffing state is now recognized with the highest F1-score value (94%), supported by balanced high precision and recall values (94% and 93% respectively), whereas Sitting, which was the first best in dataset-5 ms (95% F1-score), is the second best recognized with a F1-score value of 85%. Now, Standing is by far the most difficult to recognize (F1-score = 39%), with Lying on chest being the second worst (F1-score = 61%). From the remaining states, Trotting (F1-score = 83%) is very close to Sitting, taking the 3rd best place. Although Galloping was the most difficult to recognize in dataset-5 ms, here, it is the third most difficult.

In conclusion, the GNB algorithm performed quite well, on average, considering the simplicity of its implementation, so we had expectations of much better results from more complex algorithms.

Decision Tree. The hyperparameters of the Decision Tree model, as were tuned in the dataset-5 ms case, had values ccp_alpha = 0.001, criterion = "entropy", max_depth = 8, and max_features = 'sqrt'. This model (based on the CART algorithm) resulted in an average accuracy of 75% in predicting the dog's motion states in the test set, doing a bit better than GNB.

The situation here is similar to that of GNB as far as the best (Sniffing, F1-score = 93%) and second-best (Sitting, F1-score = 89%) recognized states are concerned. However, Galloping is the most difficult to recognize state (F1-score = 54%) now, in agreement with the dataset-5 ms case, whereas Lying on chest is the second-worst state (F1-score = 63%).

k-NN Algorithm. The hyperparameters, optimized via Grid Search, include n_neighbors = 7 and weights = 'distance'. The model achieved a prediction accuracy of 65%, much worse than the two previous models.

Although the k-NN model did not perform very well overall, it has had more balanced results. The F1-score values range from 60% to 74%. An interesting point is that it does best in the case of Galloping (F1-score = 74%), which was a problem for both GNB and DT, especially for DT. This situation is similar to that of the dataset-5 ms.

Random Forest (RF). The optimal model achieved in the dataset-5 ms case resulted in the following hyperparameter values: n_estimators = 300, max_depth = 7, min_samples_split = 2, and random_state = 42. The accuracy achieved with that model on dataset-7 ms case is 81%.

Here, there are also similar results with the first two algorithms, in the sense that the first two best recognized are the same, but with greater F1-score values (Sniffing: F1-score = 95% and Sitting: F1-score = 91%). Standing is the worst state to predict as in GNB, but with a much better score (F1-score = 65%), in contrast with the situation in the dataset-5 ms, where Standing was second-worst and Galloping was the worst. RF has also improved its success to the rest states' prediction so that it finally achieves a better overall accuracy (81%) than all previous models. This is somehow expected, given that RF is an ensemble classifier of bagging type.

Bagging Model (BM). We used the same Decision Tree model (max_depth = 5, random_state = 42) as the base classifier, and the same BaggingClassifier with n_estimators = 10, as in the case of dataset-5 ms. The prediction accuracy recorded was 69%, which is the worst result up to now. Although the first two best predicted states are Sniffing (94%) and Sitting (89%), as were in all previous models except kNN, with high scores, BM achieved low scores for the rest of the states, with Walking being the worst (30%) and Galloping being the second-worst (53%), so its overall accuracy gets the worst of all previous models. In the dataset-5 ms case, Sitting was the best recognized state (96%), followed by Standing (86%) and Trotting (84%), whereas Galloping was the worst (49%).

Stacking Model (SM). We used Decision Tree and kNN (n_neighbors = 3) as base models and Logistic Regression as a meta-classifier. SM achieved an accuracy of 81%, the same as the best (of RF) up to this point.

Here, the results are similar to the RF's ones. Again, Sniffing (94%) and Sitting (89%) are the best and the second-best predicted states, whereas Lying on chest and Standing are the two worst (71% both). In the dataset-5 ms case, Sitting was the best state (96%), followed by Trotting and Standing (89% both), with Galloping being the worst (77%).

Convolutional Neural Network (CNN). The architecture of the CNN model we constructed consists of a unique combination of Reshape, Convolutional, Max Pooling, Flatten, Dense, and Dropout layers (Figure 7a). It starts with a Reshaping layer to adapt the input shape, followed by two blocks of Conv1D and MaxPooling1D layers, which extract features and reduce spatial dimensionality while preserving important features. The Flatten layer is then used to flatten the output into a 1D vector to adapt to the subsequent fully connected layers. Two dense layers with ReLU activation and a dropout layer are employed for non-linearity and regularization. Finally, a dense layer with Softmax activation produces output probabilities for each motion state. LabelEncoder was used to convert categorical values into arithmetic ones. Utilizing the categorical crossentropy loss and Adam optimizer, supplemented by dropout and learning rate adjustments, the model's accuracy reached 89%, which is the best up to now.

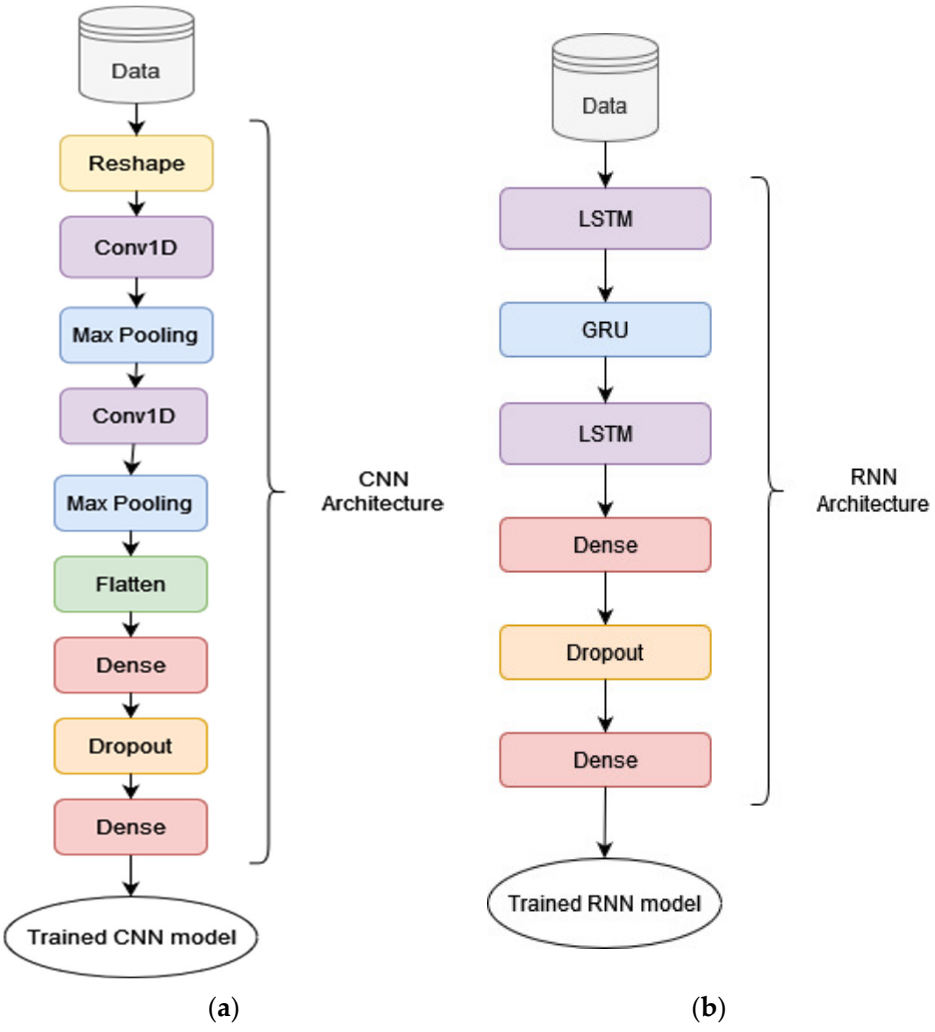


Figure 7. Architectures of the (**a**) RNN model and (**b**) CNN model.

The characteristic of CNN model is that to achieve that high accuracy, it improved its results as far as all states are concerned. It did better for all states. Again, Sniffing and Sitting are the first two best recognized states (96% and 95% respectively), with Galloping being the worst (76%). In the dataset-5 ms case, Sitting is the best state (98%), followed by Standing (94%) and Trotting (92%).

Recurrent Neural Network (RNN). The RNN model was designed using a combination of LSTM and GRU layers, as well as of a Dropout and two Dense layers (Figure 7b). It starts with an LSTM layer with 64 units, followed by a GRU layer with 32 units, both configured to return sequences to maintain temporal information. Another LSTM layer with 16 units is added for further temporal abstraction. These layers collectively extract features from sequential input data. Subsequently, a dense layer with 64 units and ReLU activation, coupled with a dropout layer for regularization, is employed for non-linearity and preventing overfitting. Finally, a dense layer with Softmax activation, equal to the number of classes, produces the output probabilities for each motion state. Adam optimizer was used during training, which includes early stopping. With settings such as epochs = 10, batch_size = 256, and a validation_split of 0.1, the model reached an accuracy of 93%.

RNN achieved the best result up to now on average and in each class separately. Sitting and Sniffing are the best states (98% and 97%, respectively), and Galloping is the worst (84%). Similarly, in the dataset-5 ms case, Sitting was the best recognized state (99%), and Galloping was the worst (86%).

Given that the best result achieved so far with the dataset-7 ms (93%, by RNN model) is worse than the best result achieved with the dataset-5 ms (94.7%, by the RNN model), we started trying new compound architectures of classifiers to improve our result. After some experimentation, we ended up with the Compound Stacking Model (CSM) and its use with an enhanced dataset.

Compound Stacking Model (CSM). The idea is to include another ensemble model as a base model in a stacking model. We selected Gradient Boosting (GB) as that ensemble model. So, our CSM architecture includes DT, GNB, and GB as base models and LR as the meta-model (see Figure 8, where the double-lined rectangle denotes an ensemble model). For each of the GNB and DT models, we used the same values for hyperparameters as those presented above. For the GB, n_estimators = 100, learning_rate = 0.1, and random_state = 42 were used. The achieved result was 91%, which was not satisfactory, being greater than that of CNN but lower than that of RNN. Here, Sniffing is the best recognized motion state (97%), with Sitting being the second-best (95%) and Galloping the worst (83%).

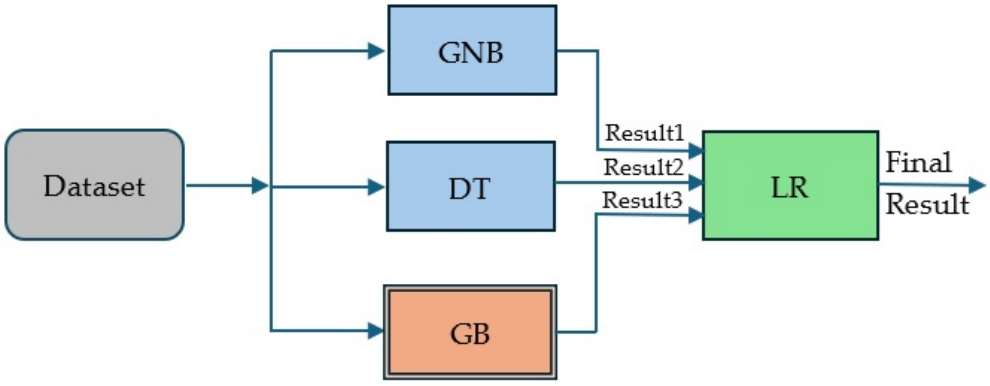


Figure 8. Architecture of the Compound Stacking Model (CSM).

Hybrid Cascading Model (HCM). The idea here came from a class of ensembles called cascading ensembles [26,27]. A cascading ensemble consists of a number of consecutively trained models, where each model is trained using the results of all previous models. In

our case, we use a two-stage cascade model, consisting of the RNN and CSM models (see Figure 9). In this case, the result of our best classifier (RNN) is used to enhance the training set of CSM. By doing this, we achieved the best of all results, an accuracy of 96.82%. Now, Sitting is the best recognized state (99%), with Sniffing being the second-best (98%) and Galloping being the worst (91%). As is obvious, all class predictions had an F1-score over 90%.

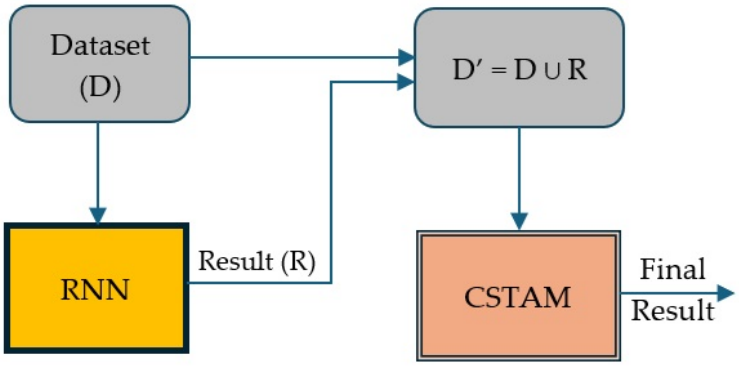


Figure 9. Architecture of the Hybrid Cascading Model (HCM).

Codes of all the above algorithms and the link for dataset-7 ms can be found at "https://github.com/ihatz/DogMotionStates/tree/MotionStatePrediction (accessed on 2 April 2025)".

5. Comparisons and Discussion

A comparison of the models used in the experiments is presented in Table 3. A basic remark is that all models used for the 5 ms dataset did worse when used for the 7 ms dataset. So, we had to devise new models to achieve better performance. Also, it is clear that the best models are, or include, deep learning models. HCM is the winner, as far as all metrics are concerned, for dataset-7 ms, achieving an accuracy of about 97%, which is almost 4% more than the next best, which is the RNN model, achieving an accuracy of 93%. CSM failed to win.

Table 3. Comparison of our models for dog motion states prediction.

Model	Accuracy		Precision		Recall		F1-Score	
	5 ms	**7 ms**	**5 ms**	**7 ms**	**5 ms**	**7 ms**	**5 ms**	**7 ms**
GNB	0.88	0.74	0.88	0.74	0.88	0.74	0.88	0.73
Decision Tree	0.84	0.75	0.83	0.75	0.84	0.75	0.83	0.75
k-NN	0.78	0.65	0.78	0.65	0.78	0.65	0.78	0.65
Random Forest	0.90	0.81	0.90	0.81	0.90	0.81	0.90	0.81
Bagging Model	0.85	0.69	0.85	0.71	0.85	0.69	0.85	0.68
Stacking Model	0.90	0.81	0.90	0.81	0.90	0.81	0.90	0.81
CNN	0.93	0.89	0.93	0.89	0.93	0.89	0.93	0.89
RNN	**0.95**	0.93	**0.95**	0.93	**0.95**	0.93	**0.95**	0.93
CSM		0.91		0.91		0.91		0.91
HCM		**0.97**		**0.97**		**0.97**		**0.97**

Notice: A bold value indicates the winner model in that column's case.

It is evident that deep learning approaches did better than (classical) ensemble methods, the difference between the best of the two groups for the 7 ms dataset (RNN and CSM) being 1.9%. On the other hand, deep learning methods require much more training time and computational resources compared to ensembles, due to their more complex architecture. Given this, as well as the small difference in accuracy and the high values

(>90%) of accuracy, it is not certain that a deep learning approach is practically preferable to an ensemble one for this problem.

In Table 4, the best and second-best, as well as the worst and second-worst, predicted features for each model are presented. It is evident that Sniffing and Sitting are the most easily predicted motion states. On the other hand, it seems that Galloping and Walking are the worst predicted features. The difficulty of predicting Galloping may be due to the small amount of data. An interesting remark is that Standing, while being the second best in most models for the dataset-5 ms case, becomes one of the worst in the dataset-7 ms case. Given that Lying on chest is, together with Standing, the second-worst predicted feature, it seems that the introduction of the Lying on chest feature in the dataset makes the distinction between them difficult. Indeed, from the confusion matrices, this is quite evident. Finally, the different behavior of the k-NN model from the rest of the models is evident.

Table 4. Best, second-best, and worst predicted features for each model.

Model	1st Best Feature		2nd Best Feature		1st Worst Feature		2nd Worst Feature	
	5 ms	7 ms	5 ms	7 ms	5 ms	7 ms	5 ms	7 ms
GNB	Sitting	Sniffing	Standing	Sitting	Galloping	Standing	Walking	Lying on chest
Decision Tree	Sitting	Sniffing	Standing	Sitting	Galloping	Galloping	Walking	Lying on chest
k-NN	Trotting	Galloping	Sitting	Trotting	Standing	Standing	Galloping	Walking
Random Forest	Sitting	Sniffing	Standing	Sitting	Galloping	Standing	Walking	Lying on chest
Bagging Model	Sitting	Sniffing	Standing	Sitting	Galloping	Walking	Walking	Galloping
Stacking Model	Sitting	Sniffing	Trotting	Sitting	Galloping	Standing	Walking	Lying on chest
CNN	Sitting	Sniffing	Standing	Sitting	Galloping	Galloping	Walking	Standing
RNN	Sitting	Sitting	Standing	Sniffing	Galloping	Galloping	Walking	Standing
CSM		Sniffing		Sitting		Galloping		Standing
HCM		Sitting		Sniffing		Galloping		Walking

In Table 5, we attempt to make a comparison of our work with other state-of-the-art works. Notice that all works in Table 5 use datasets of measures based on accelerometer and gyroscope put on dogs, except for that in [21], which uses only accelerometer data. Also, our dataset and those of [16,24] are publicly available, whereas the other datasets were created by the authors themselves. As is clear, there are differences between the datasets that existing systems use. They are due, first, to different sampling frequencies used for accelerometers and gyroscope measures; second, to different positions of those devices on the dog's body; and finally, to different labeling schemes they use. So, the comparison in Table 5 may not be precisely valid for all cases, except those using the same datasets.

Table 5. Comparison of our best models with state-of-the-art works.

Work	Dataset	States	Approach	Acc (%)
Aich et al. [14]-2019	Own	7	Deep MLP (6 layers)	96.58
Amano & Ma [19]-2021	Own	5	CNN (2Conv, 2MaxP, 2Dropout, 1Flatten, 3FC)	92.6
Kumpulainen et al. [16]-2021	Mendeley (part of)	7 (Incl. Galloping)	SVM	91.4
Muminov et al. [18]-2022	Own	6	GNB	88.0
Hussain et al. [9]-2022	Own	10	CNN (5Conv, 2Dropout, 1Flatten, 3FC)	96.85
Hussain et al. [10]-2022	Own	10	LSTM (6LSTM, 3Dropout, 3FC)	94.25

Table 5. *Cont.*

Work	Dataset	States	Approach	Acc (%)
Eerdekens et al. [21]-2022	Own	9 (incl. Sprinting)	CNN (2Conv, 1MaxP, 1Flatten, 1FC)	96.9 (10 Hz)
Marcato et al. [23]-2023	Own	5	RF Cascade	90 (F1)
Or [24]-2024	Mendeley (part of)	7 (incl. Galloping)	Encoder-FFN-GAP	**98.5–94.6 (F1)**
Ours1 [13]-2024	Kaggle (part of)	5 (incl. Galloping)	RNN (2LSTM, 1GRU, 1Dropout, 2FC)	94.7 (100 Hz)
Ours2 (HCM)-2025	Kaggle (extended part of)	7 (incl. Galloping)	RNN-CSM Cascading	**96.82** (100 Hz)

Notice: Bold values indicate the winner models.

Another remark is that only a few datasets, such as those in [16,24], concern Galloping, as ours does. Also, the dataset in [21] includes Sprinting, which is similar to Galloping. Galloping has proved to be one of the most difficult states to predict, as derived from the Table 4 analysis. This reduces the significance of the results of works [9,14]. Additionally, refs. [9,24] use more complicated deep learning architectures. Additionally, ref. [21] uses resampling of the initial data at a lower rate (from 50 Hz to 10 Hz), which proved to lead to better results, while our data was taken at a 100 Hz sampling rate. Given the above remarks, our HCM model can be considered to be at least the second-best, if not the best.

Given the good result of the cascading model, it is a motive for further investigation of this kind of model for devising lighter models achieving the same or even better results.

6. Conclusions

In this paper, we experimented with a range of machine learning algorithms, including single, ensemble, and deep learning algorithms, in solving the problem of recognizing the motion state of a dog, based on measures taken by an accelerometer and gyroscope.

Our main objective was to compare ensemble-based approaches with deep learning (DL) approaches. Results showed that the stacking model did better than other types of ensemble models and that deep learning approaches did better than ensemble models. Although we used a compound stacking model (CSM), in which one of the base models was an ensemble itself, RNN did better (by 1.78%, as far as accuracy is concerned). However, much more training time and resources are required by DL to achieve almost the same accuracy. So, the choice between an ensemble and a deep learning approach depends on this matter.

To catch up and overcome existing state-of-the-art results, we designed a two-stage cascading ensemble model, combining our RNN and CSM, thus called a hybrid cascading model (HCM). This gave a much better accuracy result (96.82%), which, compared to existing state-of-the-art results, puts our system in second place, however being a less complicated system than that in the first place.

Although we achieved a very good result, there is still room for further improvement. So, our future work will move towards finding better, perhaps simpler, system architectures that will result in even better accuracy levels.

Author Contributions: Conceptualization, G.D. and I.H.; Methodology, G.D., I.L. and I.H.; Software, G.D. and I.L.; Validation, I.H.; Formal analysis, G.D. and I.L.; Investigation, G.D. and I.L.; Data curation, I.L.; Writing—original draft, G.D. and I.H.; Writing—review & editing, I.H.; Visualization, I.L. and I.H.; Supervision, I.H. All authors have read and agreed to the published version of the manuscript.

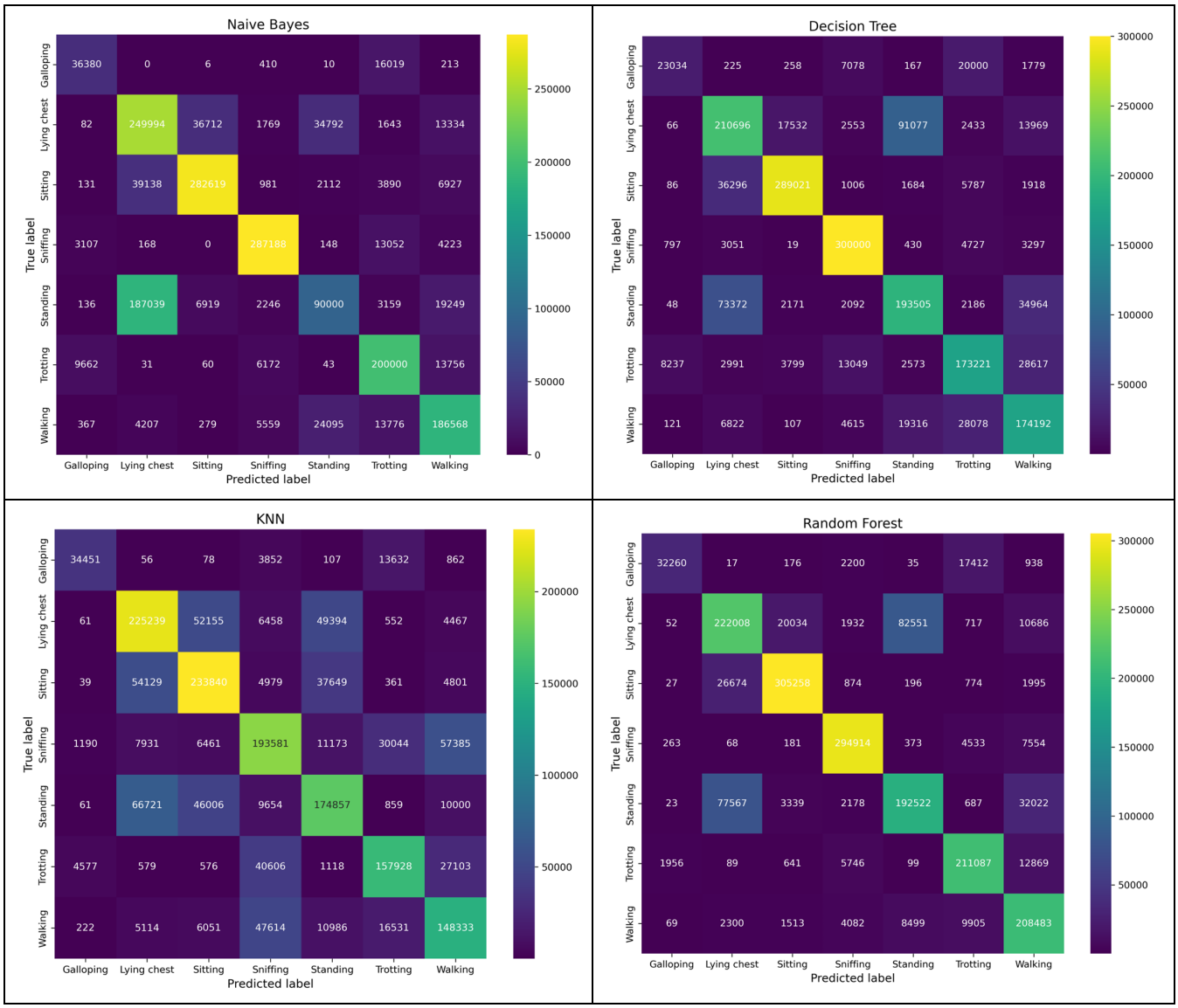
Funding: This research received no external funding.

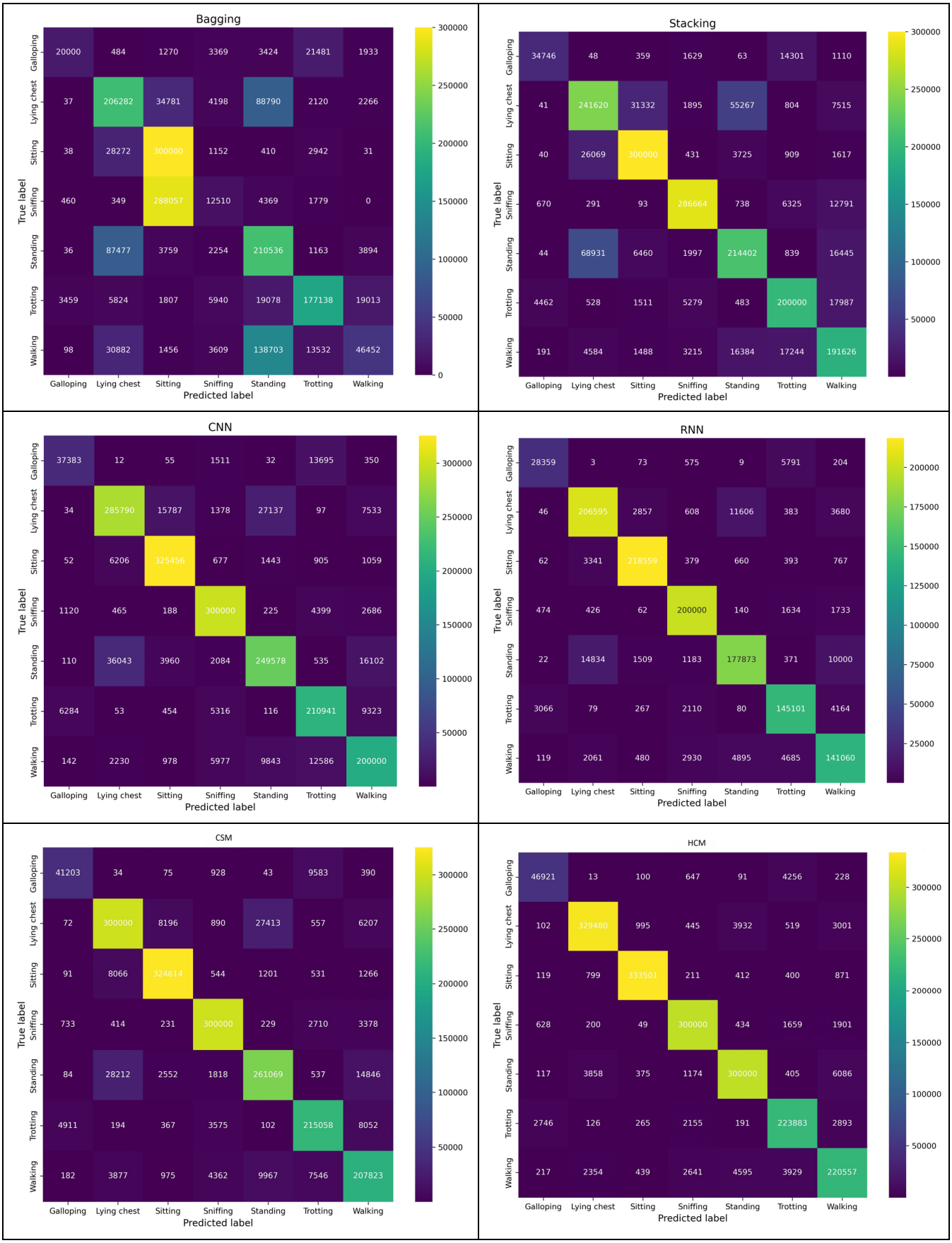
Data Availability Statement: The data presented in this study are openly available in [Github] [https://github.com/ihatz/DogMotionStates/tree/MotionStatePrediction] (accessed on 2 April 2025).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

In Appendix A, we present heatmaps of the confusion matrices of the experimented classifiers for the dataset-7 ms.




Bagging
Stacking
CNN
RNN
CSM
HCM
True label
Predicted label
Galloping
Lying chest
Sitting
Sniffing
Standing
Trotting
Walking

References

1. Hardin, A.; Schlupp, I. Using machine learning and DeepLabCut in animal behavior. *Acta Ethologica* **2022**, *25*, 125–133. [CrossRef]
2. Mao, A.; Huang, E.; Wang, X.; Liu, K. Deep learning-based animal activity recognition with wearable sensors: Overview, challenges, and future directions. *Comput. Electron. Agric.* **2023**, *211*, 108043. [CrossRef]
3. Kamat, Y.; Nasnodkar, S. Advances in technologies and methods for behavior, emotion, and health monitoring in pets. *Appl. Res. Artif. Intell. Claud Comput.* **2018**, *1*, 38–57.
4. Väätäjä, H.; Majaranta, P.; Isokoski, P.; Gizatdinova, Y.; Kujala, M.V.; Somppi, S.; Vehkaoja, A.; Vainio, O.; Juhlin, O.; Ruohonen, M.; et al. Happy dogs and happy owners: Using dog activity monitoring technology in everyday life. In Proceedings of the 5th International Conference on Animal-Computer Interaction, Atlanta, GA, USA, 4–6 December 2018; pp. 1–12. [CrossRef]
5. Rast, W.; Kimmig, S.E.; Giese, L.; Berger, A. Machine learning goes wild: Using data from captive individuals to infer wildlife behaviors. *PLoS ONE* **2020**, *15*, e0227317. [CrossRef] [PubMed]
6. Borah, B.; Saikia, R.; Das, P. Animal Motion Tracking in Forest: Using Machine Vision Technology. *Int. J. Sci. Res. Eng. Manag. (IJSREM)* **2022**, *6*, 1–8. [CrossRef]
7. Kasnesis, P.; Doulgerakis, V.; Uzunidis, D.; Kogias, D.G.; Funcia, S.I.; Gonz'alez, M.B.; Giannousis, C.; Patrikakis, C.Z. Deep learning empowered wearable-based behavior recognition for search and rescue dogs. *Sensors* **2022**, *22*, 993. [CrossRef] [PubMed]
8. Ferdinandy, B.; Gerencser, L.; Corrieri, L.; Perez, P.; Ujvary, D.; Csizmadia, G.; Miklosi, A. Challenges of machine learning model validation using correlated behaviour data: Evaluation of cross-validation strategies and accuracy measures. *PLoS ONE* **2020**, *15*, e0236092. [CrossRef] [PubMed]
9. Hussain, A.; Sikandar, A.; Abdullah Hee-Cheol, K. Activity Detection for the Wellbeing of Dogs Using Wearable Sensors Based on Deep Learning. *IEEE Access* **2022**, *10*, 53153–53163. [CrossRef]
10. Hussain, A.; Begum, K.; Armand, T.P.T.; Mozumder, A.I.; Ali, S.; Kim, H.C.; Joo, M.-I. Long Short-Term Memory (LSTM)-Based Dog Activity Detection Using Accelerometer and Gyroscope. *Appl. Sci.* **2022**, *12*, 9427. [CrossRef]
11. Hatzilygeroudis, I.; Prentzas, J. AI Approaches for the Prognosis of the Survival (or Not) of Patients with Bone Metastases. In Proceedings of the IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI), Washington, DC, USA, 1–3 November 2021; pp. 1353–1357. [CrossRef]
12. Troussas, C.; Krouska, A.; Virvou, M. Evaluation of ensemble-based sentiment classifiers for Twitter data. In Proceedings of the 7th International Conference on Information, Intelligence, Systems & Applications (IISA), Chalkidiki, Greece, 13–15 July 2016; pp. 1–6. [CrossRef]
13. Davoulos, G.; Lalakou, I.; Hatzilygeroudis, I. Recognition of Dog Motion States: Ensemble vs Deep Learning Models. In Proceedings of the 15th International Conference on Information, Intelligence, Systems & Applications (IISA), Chania Crete, Greece, 17–19 July 2024; pp. 1–8. [CrossRef]
14. Aich, S.; Chakraborty, S.; Sim, J.-S.; Jang, D.-J.; Kim, H.-C. The Design of an Automated System for the Analysis of the Activity and Emotional Patterns of Dogs with Wearable Sensors Using Machine Learning. *Appl. Sci.* **2019**, *9*, 4938. [CrossRef]
15. Chambers, R.D.; Yoder, N.C.; Carson, A.B.; Junge, C.; Allen, D.E.; Prescott, L.M.; Bradley, S.; Wymore, G.; Lloyd, K.; Lyle, S. Deep Learning Classification of Canine Behavior Using a Single Collar-Mounted Accelerometer: Real-World Validation. *Animals* **2021**, *11*, 1549. [CrossRef] [PubMed]
16. Kumpulainen, P.; Cardo, A.V.; Somppi, S.; Tornqvist, H.; Vaataja, H.; Majaranta, P.; Gizatdinova, Y.; Antink, C.H.; Surakka, V.; Kujala, M.V.; et al. Dog behaviour classification with movement sensors placed on the harness and the collar. *Appl. Anim. Behav. Sci.* **2021**, *241*, 105393. [CrossRef]
17. Vehkaoja, A.; Somppi, S.; Törnqvist, H.; Cardó, A.V.; Kumpulainen, P.; Väätäjä, H.; Majaranta, P.; Surakka, V.; Kujala, M.V.; Vainio, O. Description of Movement Sensor Dataset for Dog Behavior Classification. *Data Brief* **2022**, *40*, 107822. [CrossRef] [PubMed]
18. Muminov, A.; Mukhiddinov, M.; Cho, J. Enhanced Classification of Dog Activities with Quaternion-Based Fusion Approach on High-Dimensional Raw Data from Wearable Sensors. *Sensors* **2022**, *22*, 9471. [CrossRef] [PubMed]
19. Amano, R.; Ma, J. Recognition and Change Point Detection of Dogs' Activities of Daily Living Using Wearable Devices. In Proceedings of the 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, Virtual, 25–28 October 2021; pp. 693–699. [CrossRef]
20. Kim, J.; Moon, N. Dog Behavior Recognition Based on Multimodal Data from a Camera and Wearable Device. *Appl. Sci.* **2022**, *12*, 3199. [CrossRef]
21. Eerdekens, A.; Callaert, A.; Deruyck, M.; Martens, L.; Joseph, W. Dog's Behaviour Classification Based on Wearable Sensor Accelerometer Data. In Proceedings of the 5th Conference on Cloud and Internet of Things (CIoT-22), Marrakech, Morocco, 28–30 March 2022; pp. 226–231.
22. Kim, H.; Moon, N. TN-GAN-Based Pet Behavior Prediction through Multiple-Dimension Time-Series Augmentation. *Sensors* **2023**, *23*, 4157. [CrossRef] [PubMed]

23. Marcato, M.; Tedesco, S.; O'Mahony, C.; O'Flynn, B.; Galvin, P. Machine learning based canine posture estimation using inertial data. *PLoS ONE* **2023**, *18*, e0286311. [CrossRef] [PubMed]
24. Or, B. Transformer Based Dog Behavior Classification with Motion Sensors. *IEEE Sens. J.* **2024**, *24*, 33816–33825. [CrossRef]
25. Chatzilygeroudis, K.; Hatzilygeroudis, I.; Perikos, I. Machine Learning Basics. In *Intelligent Computing for Interactive System Design: Statistics, Digital Signal Processing, and Machine Learning in Practice*; ACM: New York, NY, USA, 2021; pp. 143–193. [CrossRef]
26. Garcıa-Pedrajas, N.; Ortiz-Boyer, D.; del Castillo-Gomariz, R.; Hervas-Martınez, C. Cascade Ensembles. In *Computational Intelligence and Bioinspired Systems. IWANN 2005*; Cabestany, J., Prieto, A., Sandoval, D.F., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3512. [CrossRef]
27. De Zarzà, I.; de Curtò, J.; Hernández-Orallo, E.; Calafate, C.T. Cascading and Ensemble Techniques in Deep Learning. *Electronics* **2023**, *12*, 3354. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.