Physiology & Behavior 302 (2025) 115105

Contents lists available at ScienceDirect

Physiology & Behavior

journal homepage: www.elsevier.com/locate/physbeh

Research paper

Identification of human behavior in accelerometer data from interactive toys by applying AutoML

Eddy Sánchez-DelaCruz, Cecilia-Irene Loeza-Mejía *, Irahan-Otoniel José-Guzmán, Mirta Fuentes-Ramos

Artificial Intelligence Lab., Tecnológico Nacional de México/Instituto Tecnológico Superior de Misantla, Misantla, Veracruz, Mexico

ARTICLE INFO

Dataset link: https://doi.org/10.7910/DVN/FHOO0Q

Keywords:
Behavior
Machine learning
Human–machine interaction
AutoML
AutoWEKA

ABSTRACT

Background: Human behavior is closely tied to our identities, cultures, and illnesses, and is therefore highly relevant to social, commercial, and medical studies. Analyzing interactions between people or between people and items is a method for studying behavior. In this work, we analyze pre-recorded accelerometer data from interactions with embedded sensors to classify 8,946 behavior records from five classes: *drop*, *hit*, *pickup*, *shake*, and *throw*.

Methods: We evaluated multiple machine learning algorithms—Bayes Network, Multinomial Logistic Regression, Multi-layer Perceptron, Naïve Bayes, and Repeated Incremental Pruning to Produce Error Reduction (RIPPER). Also, an AutoML approach was applied for automated model and hyperparameter selection.

Results: AutoML outperformed traditional classifiers, achieving a precision of 94.4% and a receiver operating characteristic (ROC) area of 0.992 were obtained.

Conclusion: These findings confirm AutoML's effectiveness in accurately identifying human behaviors from accelerometer data in interactive toys.

1. Introduction

Human behavior is a complex construct shaped by individual and social traits influenced by heredity, social skills, phenomenology, and diseases [1]. Behavioral disorders, often linked to premorbid functioning, play a crucial role in assessing and treating individuals with brain injuries [1,2]. A deeper understanding of human behavior enables the development of clinically efficient and cost-effective interventions [1] and enhances interactions in domains such as human–computer and human–robot interfaces [3,4].

Traditionally, behavioral analysis has relied on observational techniques, which are often labor-intensive and inefficient for research and clinical settings [5–7]. The integration of computational tools has revolutionized this field, enabling automated behavior assessment through facial expressions, head movements, gestures, and object interactions [3,8,9]. Advancements in healthcare, the development of less invasive techniques, improved visual inspection, and reduced economic losses are among the key benefits of integrating behavioral analysis into technological frameworks [10–12]. Machine learning and computer vision techniques, in particular, have proven highly effective in behavioral analysis. For instance, Usman et al. [9] configured a smart chair for posture and object interaction analysis, achieving 98.75% accuracy with a multilayer perceptron. Similarly, Alban et al. [13] performed a binary classification of behaviors of autistic children, they achieved 97% recall by implementing a multilayer perceptron. These works highlight the potential of machine learning algorithms in extracting valuable insights from behavioral data.

Despite these advancements, research on human interactions with interactive toys through machine learning remains limited [14]. Given the growing role of these toys in child development [15], software games [16], education [17], and therapy — particularly for children with autism [18] and other disorders — understanding these interactions is essential. Advancing human–robot interaction through intelligent, personalized interfaces can enhance user engagement and learning experiences.

Machine learning approaches have demonstrated their ability to extract knowledge from data. However, model performance is highly dependent on hyperparameter selection [19]. The same algorithm applied to the same problem can produce varying results depending on the chosen hyperparameters, leading to multiple possible outcomes [19,20]. Traditionally, hyperparameter tuning relies on trial-and-error

* Corresponding author.
E-mail address: cecilialoeza@yahoo.com (C.-I. Loeza-Mejía).
URL: https://www.cecilialoeza.com (C.-I. Loeza-Mejía).

https://doi.org/10.1016/j.physbeh.2025.115105
Received 1 May 2025; Received in revised form 7 July 2025; Accepted 15 September 2025
Available online 22 September 2025

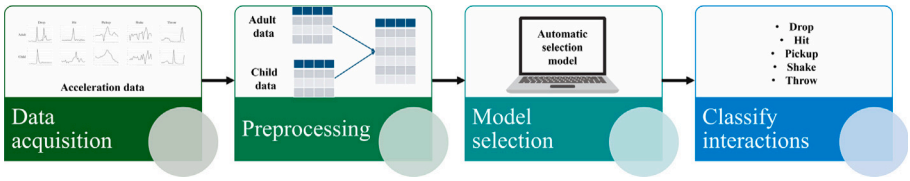

Fig. 1. Methodology employed in this study.

experimentation to determine the optimal configuration [19]. To address these challenges, AutoML (Automated Machine Learning) has emerged as a promising subfield aimed at automating hyperparameter selection, minimizing human intervention, and enhancing model performance. Based on the analyzed studies, a niche opportunity was identified in examining human behavior within human–robot interactions, particularly using accelerometer data to capture behavioral responses to interactive toys. In this study, we evaluated several machine learning algorithms — including Bayes Network, Multinomial Logistic Regression, Multi-layer Perceptron, Naïve Bayes, and RIPPER — comparing their performance against an AutoML approach based on Thornton et al. [20–23] to classify human behaviors toward interactive toys. Using a dataset of 8946 instances of the datasets recognition of aggressive interactions of children toward toys [24].

The remainder of the article is structured as follows: Section 2 includes materials and methods, Section 3 elaborates on the results and analysis, and finally, Section 4 presents conclusions and avenues for further research.

2. Material and methods

The methodology employed in this study (see Fig. 1) follows a structured approach to data acquisition, preprocessing, and model selection to classify aggressive interactions of children toward toys based on acceleration data. First, publicly available datasets were obtained and merged, selecting specific interaction classes for analysis. Subsequently, preprocessing steps were applied to transform categorical variables into numerical representations suitable for machine learning models. No data augmentation techniques were applied during preprocessing, in order to preserve the integrity and original distribution of the collected data. Several traditional machine learning algorithms — such as Bayes Network, Multinomial Logistic Regression, Multi-layer Perceptron, Naïve Bayes, and RIPPER — were evaluated under the same conditions to compare their classification performance. However, only the results of the AutoML approach are presented in detail, as it demonstrated the highest overall performance. Cross-validation was applied across all algorithms to ensure a robust performance evaluation. The following subsections describe each step in detail.

2.1. Data acquisition

Three datasets (files) were acquired: 0_adult_train_dataset.csv, 0_adult_test_dataset.csv, and 0_children_test_dataset.csv of recognition of aggressive interactions of children toward toys [24], which are publicly available on Harvard Dataverse.[1] These three datasets were selected based on behaviors that were collected during human–robot interaction utilizing accelerometers[2] mounted on a Raspberry Pi3[3] and embedded in toys. The datasets store magnitudes of accelerations produced by different behaviors of 6 adults and 4 children toward three toys (a stuffed panda, a stuffed robot, and an excavator). Each observation of the datasets includes acceleration data, participant number, toy, and behavior.

[1] https://doi.org/10.7910/DVN/FHOO0Q
[2] LSM9DS1, STMicroelectronics, Switzerland.
[3] Pi3 Model B+, Raspberry Pi Foundation, United Kingdom.

Table 1
Total number of records by class.

	Class	Rows
	Drop	1314
	Hit	1992
	Pickup	1803
	Shake	2153
	Throw	1684
Total		8946

Table 2
Encoding of nominal columns.

Column	Original value	Value assigned
Participant	Adult1	0
	Adult2	0
	Adult3	0
	Adult4	0
	Adult5	0
	Adult6	0
	Child1	1
	Child2	1
	Child3	1
	Child4	1
Robot	excavator	0
	Panda	1
	robot_toy	2

Table 3
Dataset adapted from [24] (fragment).

a1	a2	…	a25	Participant	Robot	Behavior
0.487691278	0.225038251	…	1.030042075	0	0	drop
0.969817654	0.981271725	…	0.977435205	0	0	hit
0.821271579	1.525458072	…	0.590929526	0	1	pickup
1.178780354	1.420366315	…	1.216250532	0	2	shake
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
1.181332641	1.334889874	…	7.879113183	1	2	throw

To perform the data preprocessing, the three mentioned datasets were merged using the records of the classes: *drop*, *hit*, *pickup*, *shake*, and *throw*. Total records by class and overall total are shown in Table 1. Subsequently, the nominal column values (i.e. Participant, Robot) were transformed to a numeric format as shown in Table 2. Table 3 shows a fragment of the dataset used to train the AutoML model. In Fig. 2, graphs of samples of accelerations for each of the classes are shown.

2.2. Automated machine learning

In this study, two experiments were performed implementing the model selection method of Thornton et al. [20–22], which automatically searches within a set of algorithms $\mathcal{A}$ for the appropriate hyperparameters to map a training set $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ of data points $d_i = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}$ in a function $f : \mathcal{X} \to \mathcal{Y}$. The algorithm $A^* \in \mathcal{A}$ with the optimal generalization performance is determined, which is estimated by dividing $\mathcal{D}$ into the disjoint subsets $\mathcal{D}^{(i)}_{train}$ and $\mathcal{D}^{(i)}_{valid}$. It is applied A^* to $\mathcal{D}^{(i)}_{train}$ and assesses its performance in $\mathcal{D}^{(i)}_{valid}$. The sampling criterion k-fold cross-validation is used to separate training

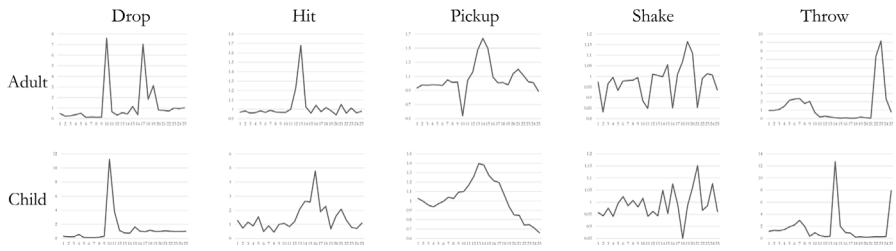

Fig. 2. Samples of accelerations for each of the classes.

data in k partitions of equal size, i.e.: $\mathcal{D}^{(1)}_{valid}, \ldots, \mathcal{D}^{(k)}_{valid}$, and is obtained $\mathcal{D}^{(i)}_{train} = \mathcal{D} \setminus \mathcal{D}^{(i)}_{valid}$ for $i = \{1, \ldots, k\}$. In each subset the misclassification rate $\mathcal{L}$. Finally, the problem of model selection is written as follows:

$$A^* \in [A \in \mathcal{A}] argmin \frac{1}{k} \cdot \sum_{i=1}^{k} \mathcal{L}(A, \mathcal{D}^{(i)}_{train}, \mathcal{D}^{(i)}_{valid}) \quad (1)$$

The method was configured with a batch size of 100, a memory limit of 1024 bytes, a metric error rate, and a seed of 123. Two experimental scenarios were considered: Experiment-A (Exp-A) with a 15 min time limit and Experiment-B (Exp-B) with a 30 min limit.[4] These time constraints were selected to assess performance under different computational limits and analyze their impact on model behavior. The dataset was partitioned using cross-validation to ensure robust evaluation across experiments.

In Exp-A the algorithm that performed better was random forest [26]. Whereas, in Exp-B, the best algorithm was the random subspace method [27]. Both algorithms are based on a decision tree-based classifier which is one of the best-performing algorithms for building assemblies [28], in addition to being simple and functional in various data types for classification [26,29–32].

- *Random forest* is based on bagging and random subspace method. Random forest generates k tree-structured classifiers $\{h(x, \Theta_k), k = 1, \ldots\}$ where x is an input vector and Θ_k are independent vectors that create a prediction [26,29]. Random forest algorithm is shown in Algorithm 1.
- *Random subspace method* constructs a decision forest ensemble [27], having S training sample with X_j p-dimensional vectors, that is $X_j = (x_{j1}, x_{j2}, \ldots, x_{jp})$. Random subspace method automatically selects $p *$ features where $p *< p$ [28]. Random subspace algorithm is shown in Algorithm 2.

Algorithm 1: Random forest [26,33,34]

Input : training set T, number of trees m, number of random levels k
Output: *RF*, a set of grown trees
Initialization *RF*
for $i = 1$ *to* m **do**
  T' ← *boostrap(T)*
  Tree ← *trainDT (T'k)*
  add *Tree* to *RF*

2.3. Validation metrics

The metrics used were the multiclass confusion matrix (see Table 4) and ROC area [37]. The confusion matrix generates a relationship between the true class values and the class labels predicted by a machine learning model. The confusion matrix includes the values true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Based on the confusion matrix, are calculated the metrics TP rate or recall (TPR), FP rate (FPR), precision or positive predictive value (PPV), and F_1-Score for each of the classes C_i [38].

Algorithm 2: Random subspace method [27,28,35,36]

Input : training set S, number of subspaces B, dimension of subspaces $p*$
Output: ensemble E
$E \leftarrow 0$
for $i = 1$ *to* B **do**
  $\tilde{S}^i \leftarrow$ SelectRandomSubspace($\tilde{S}$,p*)
  $C^i \leftarrow ConstructClassifier(\tilde{S}^i)$
  $E \leftarrow E \cup \{C^i\}$

Table 4
Matrix of confusion multiclass [38].

		Predicted class			
		C_1	C_2	…	C_N
True class	C_1	$C_{1,1}$	FP	…	$C_{1,N}$
	C_2	FN	TP	…	FN
	…	…	…	…	…
	C_N	$C_{N,1}$	FP	…	$C_{N,N}$

$$TPR(C_i) = \frac{TP(C_i)}{TP(C_i) + FN(C_i)} \quad (2)$$

$$FPR(C_i) = \frac{FP(C_i)}{FP(C_i) + TN(C_i)} \quad (3)$$

$$PPV(C_i) = \frac{TP(C_i)}{TP(C_i) + FP(C_i)} \quad (4)$$

$$F_i(C_i) = \frac{2 \cdot TPR(C_i) \cdot PPV(C_i)}{TPR(C_i) + PPV(C_i)} \quad (5)$$

3. Results

Experiments were carried out on a computer with the operating system Windows 10 64 bits, Intel(R) Core(TM) i7-1065G7 CPU @ 1.30 GHz (8 CPUs), 1.5 GHz, and 12 GB RAM. The Thornton et al. method [21,22] was implemented using the library Auto-Weka 2.6.4 version available on Weka 3.8.6.[5]

Two experiments were carried out by applying AutoML: i) Exp-A and ii) Exp-B. Table 5 shows the confusion matrix obtained in Exp-A, in which it was observed that for the class *drop*, 1298 behaviors' records are classified (identified) correctly, and 16 are not recognized. In the class *hit*, 1774 behaviors' records are classified correctly, and 218 are not recognized. In the class *pickup*, 1593 are classified correctly, and 210 are not recognized. In the class *shake*, 2072 are classified correctly, and 81 are not recognized. While, in the class *throw*, 1635 are classified correctly, and 49 are not recognized.

[4] The "restriction" refers to the maximum duration allowed before interrupting the training process of a learning algorithm [25].

[5] https://www.cs.waikato.ac.nz/ml/weka/index.html

Table 5
Confusion matrix of Exp-A.

classified as	*a*	*b*	*c*	*d*	*e*
drop = *a*	1298	0	2	14	0
hit = *b*	0	1774	175	42	1
pickup = *c*	0	196	1593	12	2
shake = *d*	7	28	35	2072	11
throw = *e*	3	3	0	43	1635

Table 6
Confusion matrix of Exp-B.

classified as	*a*	*b*	*c*	*d*	*e*
drop = *a*	1294	0	1	19	0
hit = *b*	0	1794	153	44	1
pickup = *c*	0	160	1627	14	2
shake = *d*	5	13	26	2094	15
throw = *e*	4	0	0	46	1634

Table 7
Detailed metrics by class of Exp-A.

	Class	TPR	FPR	PPV	F_1	ROC area
	drop	**0.988**	**0.001**	**0.992**	**0.990**	**1.000**
	hit	0.891	0.033	0.887	0.889	0.984
	pickup	0.884	0.030	0.883	0.883	0.984
	shake	0.962	0.016	0.949	0.956	0.994
	throw	0.971	0.002	**0.992**	0.981	0.997
Weighted Avg.		0.936	0.018	0.936	0.936	0.991

Table 8
Detailed metrics by class of Exp-B.

	Class	TPR	FPR	PPV	F_1	ROC area
	drop	**0.985**	**0.001**	**0.993**	**0.989**	**1.000**
	hit	0.901	0.025	0.912	0.906	0.986
	pickup	0.902	0.025	0.900	0.901	0.986
	shake	0.973	0.018	0.945	0.958	0.995
	throw	0.970	0.002	0.989	0.980	0.998
Weighted Avg.		0.944	0.016	0.944	0.944	0.992

Table 9
Performance metrics comparison of traditional machine learning algorithms and AutoML (Exp-B).

Algorithm	TPR	FPR	PPV	F_1	ROC area
Bayes Network	0.857	0.041	0.861	0.857	0.976
Multinomial logistic regression	0.854	0.04	0.857	0.855	0.971
Multi-layer perceptron	0.882	0.032	0.883	0.882	0.977
Naïve Bayes	0.735	0.071	0.752	0.721	0.92
RIPPER	0.888	0.031	0.888	0.888	0.966
AutoML (Exp-B)	**0.944**	**0.016**	**0.944**	**0.944**	**0.992**

Table 6 displays Exp-B results, in which it was observed that for the class *drop*, 1294 behaviors' records are classified correctly and 20 are not recognized. In the class *hit*, 1794 behaviors' records are classified correctly, and 198 are not recognized. In the class *pickup*, 1627 are classified correctly, and 176 are not recognized. In the class *shake*, 2094 are classified correctly, and 59 are not recognized. While, in the class *throw*, 1634 are classified correctly, and 50 are not recognized.

On the other hand, Tables 7 and 8 expose the metrics obtained by each class in each experiment. In both tables, they stand out in **bold** the best results. The best ***overall ranking*** rate was achieved in Exp-B, with TPR of 0.944, FPR of 0.016, PPV of 0.944, F_1 of 0.944, and ROC area of 0.992.

In addition to AutoML, several traditional machine learning algorithms were evaluated under the same conditions to compare classification performance. Table 9 summarizes these results, where AutoML outperformed Bayes Network, Multinomial Logistic Regression, Multi-layer Perceptron, Naïve Bayes, and RIPPER across all metrics.

3.1. Discussion

This study conducted a comparative analysis between several traditional machine learning algorithms — Bayes Network, Multinomial Logistic Regression, Multi-layer Perceptron, Naïve Bayes, and RIPPER — and the AutoML approach. All models were evaluated under the same experimental conditions to ensure a fair comparison. As shown in Table 9, the AutoML model obtained the highest overall performance across all evaluation metrics, particularly in Exp-B, which allowed a longer training duration. The best-performing traditional algorithm was **RIPPER**, with a TPR, PPV, F_1-score of 0.888 and a ROC Area of 0.966, followed closely by the **Multi-layer Perceptron**, which obtained a TPR and F_1-score of 0.882, PPV of 0.883, and a ROC Area of 0.977. These results demonstrate that rule-based learners and neural networks are competitive options when properly configured.

In both AutoML experiments, the best classification was achieved in the class *drop* (Tables 7 and 8), achieving an optimal ROC area value of 1.0 (100%). This confirms that Random Forest and Random Subspace methods perform well for classification tasks, as supported by the literature [26,29,30].

A key observation is that Exp-B consistently outperformed Exp-A across all metrics (TPR, PPV, F_1, and ROC area). The improvement is particularly evident in the classes *hit* and *pickup*, where the true positive rate (TPR) increased from 0.891 to 0.901 and from 0.884 to 0.902, respectively. This suggests that the additional training time in Exp-B allowed the AutoML model to refine decision boundaries more effectively.

Interestingly, classification performance does not appear to correlate with the number of behavior records. Despite having fewer samples, the class *drop* achieved the highest accuracy, while the class *hit*, with a larger dataset, showed more misclassifications. This reinforces the idea that feature quality and model training conditions play a more crucial role than sheer data volume.

Moreover, no signs of overfitting or overtraining were observed in the experiments, which were consistent across both the training and validation sets, and no significant performance degradation was detected in unseen data.

These findings suggest that longer training time can enhance model performance, particularly for differentiating similar behaviors. Future work should explore whether further extending the training duration continues to yield improvements or if diminishing returns are observed. Additionally, testing these models on unseen datasets would help assess their generalization capabilities. On the other hand, applying this approach in real-world scenarios — such as domestic, therapeutic, or educational environments involving interactive toys — requires consideration of practical constraints. For instance, it may be relevant to evaluate whether the model can be embedded directly into the toy to enable real-time reactions to specific behaviors (e.g., stopping an activity when the toy is hit). Alternatively, the data could be stored locally or in the cloud for subsequent analysis, aimed at automatically identifying specific behavioral patterns—particularly those that reflect

undesirable or atypical actions. This would enable diagnostic use, supporting caregivers or therapists in detecting early signs of behavioral issues or monitoring the user's developmental progress.

In either case, it is necessary to address aspects such as computational limitations, energy consumption, prediction latency, and data privacy to ensure reliable performance in dynamic environments. The current AutoML pipeline may require adaptations or simplifications, and lightweight models should be explored to meet the constraints of embedded systems or mobile applications. These considerations are essential for transitioning from experimental validation to practical solutions with real-world impact.

4. Conclusion and future works

Behavioral analysis has become a crucial area of study due to its significant implications in diagnosing diseases, medical treatment, and human–computer interaction. In this study, we compared a set of traditional machine learning algorithms — such as Bayes Network, Multinomial Logistic Regression, Multi-layer Perceptron, Naïve Bayes, and RIPPER — with an AutoML approach. The results demonstrated that an AutoML approach achieved superior performance across key evaluation metrics of 0.944 and a ROC Area of 0.992, surpassing all traditional models tested. This enhancement in performance demonstrates AutoML's capacity to streamline the model development process while improving classification accuracy, thereby reinforcing its potential to advance behavior recognition systems in human–robot interaction contexts.

- Conduct behavioral case studies involving participants with different medical conditions to observe their behavioral responses and evaluate the model's potential as a diagnostic support tool.
- It would be of great interest to expand behavior analysis and integrate it with ambient intelligence, internet of things, and fog computing technologies.
- Evaluate the performance of the proposed models on actual embedded hardware integrated into robotic toys, in order to assess their feasibility and efficiency under real-world operational constraints.
- Exploring the possibility of implementing this method directly within the toy, enabling it to appropriately respond to specific user behaviors in real time.
- Consider storing data locally or remotely for subsequent analysis, with the goal of automatically identifying specific behaviors, particularly those that may be undesirable.

CRediT authorship contribution statement

Eddy Sánchez-DelaCruz: Writing – original draft, Methodology, Investigation, Data curation, Conceptualization. **Cecilia-Irene Loeza-Mejía:** Writing – review & editing, Writing – original draft, Supervision, Project administration, Investigation, Formal analysis. **Irahan-Otoniel José-Guzmán:** Writing – original draft, Methodology, Investigation. **Mirta Fuentes-Ramos:** Writing – original draft, Methodology, Investigation.

Declaration of competing interest

The authors declare that they have no conflict of interest.

Acknowledgments

Secretaría de Ciencia, Humanidades, Tecnología e Innovación (Secihti)

Data availability

https://doi.org/10.7910/DVN/FHOO0Q.

References

[1] H.E. Jacobs, Essentials of behavior analysis, in: Behavior Analysis Guidelines and Brain Injury Rehabilitation: People, Principles, and Programs, Aspen Publishers, Gaithersburg, Maryland, 1993, pp. 3–11.
[2] H.E. Jacobs, Behavior analysis guidelines and brain injury rehabilitation: People, principles, and programs., Aspen Publishers, Gaithersburg, Maryland, 1993.
[3] T. Baltrusaitis, A. Zadeh, Y.C. Lim, L.-P. Morency, Openface 2.0: Facial behavior analysis toolkit, in: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition, FG 2018, IEEE, 2018, pp. 59–66.
[4] N. Efthymiou, P.P. Filntisis, G. Potamianos, P. Maragos, Visual robotic perception system with incremental learning for child–robot interaction scenarios, Technologies 9 (4) (2021) 86.
[5] R. Arablouei, L. Currie, B. Kusy, A. Ingham, P.L. Greenwood, G. Bishop-Hurley, In-situ classification of cattle behavior using accelerometry data, Comput. Electron. Agric. 183 (2021) 106045.
[6] K. Zhang, D. Li, J. Huang, Y. Chen, Automated video behavior recognition of pigs using two-stream convolutional networks, Sensors 20 (4) (2020) 1085.
[7] C. Segalin, J. Williams, T. Karigo, M. Hui, M. Zelikowsky, J.J. Sun, P. Perona, D.J. Anderson, A. Kennedy, The mouse action recognition system (MARS) software pipeline for automated analysis of social behaviors in mice, Elife 10 (2021) e63720.
[8] J. de Wit, E. Krahmer, P. Vogt, Introducing the NEMO-lowlands iconic gesture dataset, collected through a gameful human–robot interaction, Behav. Res. Methods 53 (3) (2021) 1353–1370.
[9] M. Usman, Z. Noor, I. Farooq, A. Arsalan, M. Ehatisham-ul Haq, A. Raheel, A smart chair design for recognizing human-object interactions using pressure sensors, in: 2020 IEEE 23rd International Multitopic Conference, INMIC, IEEE, 2020, pp. 1–6.
[10] A. Brahim, B. Malika, A. Rachida, L. Mustapha, D. Mehammed, L. Mourad, Dairy cows real time behavior monitoring by energy-efficient embedded sensor, in: 2020 Second International Conference on Embedded & Distributed Systems, EDiS, IEEE, 2020, pp. 21–26.
[11] D. Pavlovic, M. Czerkawski, C. Davison, O. Marko, C. Michie, R. Atkinson, V. Crnojevic, I. Andonovic, V. Rajovic, G. Kvascev, et al., Behavioural classification of cattle using neck-mounted accelerometer-equipped collars, Sensors 22 (6) (2022) 2323.
[12] J.A. Vázquez Diosdado, Z.E. Barker, H.R. Hodges, J.R. Amory, D.P. Croft, N.J. Bell, E.A. Codling, Classification of behaviour in housed dairy cows using an accelerometer-based activity monitoring system, Anim. Biotelemetry 3 (1) (2015) 1–14.
[13] A.Q. Alban, M. Ayesh, A.Y. Alhaddad, A.K. Al-Ali, W.C. So, O. Connor, J.-J. Cabibihan, Detection of challenging behaviours of children with autism using wearable sensors during interactions with social robots, in: 2021 30th IEEE International Conference on Robot & Human Interactive Communication, RO-MAN, IEEE, 2021, pp. 852–857.
[14] A.Y. Alhaddad, J.-J. Cabibihan, A. Bonarini, Influence of reaction time in the emotional response of a companion robot to a child's aggressive interaction, Int. J. Soc. Robot. 12 (6) (2020) 1279–1291.
[15] Y. Fan, D.K. Chong, Y. Li, Beyond play: a comparative study of multi-sensory and traditional toys in child education, in: Frontiers in Education, vol. 9, Frontiers Media SA, 2024, 1182660.
[16] R. Luckin, D. Connolly, L. Plowman, S. Airey, Children's interactions with interactive toy technology, J. Comput. Assist. Learn. 19 (2) (2003) 165–176.
[17] P. Gondaliya, Recommendations on how educational toys can help preschoolers to improve their social skills, Educ. Adm.: Theory Pr. 30 (3) (2024) 975–979.
[18] R.C. Cañete, A. Picardo, P. Trueba, Y. Torres, E. Peralta, A new multi-criteria decision-making approach for the design and selection of materials and manufacturing processes of toys for children with autism, Mater. Today Commun. (2024) 109709.
[19] R.G. Mantovani, A.L.D. Rossi, E. Alcobaça, J.C. Gertrudes, S.B. Junior, A.C.P.d.F. de Carvalho, Rethinking default values: a low cost and efficient strategy to define hyperparameters, 2020, arXiv preprint arXiv:2008.00025.
[20] C. Thornton, F. Hutter, H.H. Hoos, K. Leyton-Brown, et al., Auto-weka: Automated selection and hyper-parameter optimization of classification algorithms, 2012, CoRR, Abs/1208.3719.
[21] C. Thornton, F. Hutter, H.H. Hoos, K. Leyton-Brown, Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2013, pp. 847–855.
[22] L. Kotthoff, C. Thornton, H.H. Hoos, F. Hutter, K. Leyton-Brown, Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA, J. Mach. Learn. Res. 17 (2016) 1–5.

[23] L. Kotthoff, C. Thornton, H.H. Hoos, F. Hutter, K. Leyton-Brown, Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA, J. Mach. Learn. Res. 18 (25) (2017) 1–5.
[24] A.Y. Alhaddad, J.-J. Cabibihan, A. Bonarini, Datasets for recognition of aggressive interactions of children toward robotic toys, Data Brief 34 (2021) 106697.
[25] C. Thornton, Auto-WEKA: Combined Selection and Hyperparameter Optimization of Supervised Machine Learning Algorithms (Ph.D. thesis), University of British Columbia, 2014.
[26] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[27] T.K. Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Mach. Intell. 20 (8) (1998) 832–844, URL http://citeseer.ist.psu.edu/ho98random.html.
[28] P. Panov, S. Džeroski, Combining bagging and random subspaces to create better ensembles, in: International Symposium on Intelligent Data Analysis, Springer, 2007, pp. 118–129.
[29] A.R. Panhalkar, D.D. Doye, A novel approach to build accurate and diverse decision tree forest, Evol. Intell. (2022) 1–15.
[30] A. Criminisi, J. Shotton, E. Konukoglu, et al., Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning, Found. Trends® Comput. Graph. Vis. 7 (2–3) (2012) 81–227.
[31] V.F. Rodriguez-Galiano, B. Ghimire, J. Rogan, M. Chica-Olmo, J.P. Rigol-Sanchez, An assessment of the effectiveness of a random forest classifier for land-cover classification, ISPRS J. Photogramm. Remote Sens. 67 (2012) 93–104.
[32] J.H. Moore, Mining patterns of epistasis in human genetics, in: Biological Data Mining, Chapman and Hall/CRC, New York, 2010, pp. 207–224.
[33] V. Sazonau, Implementation and Evaluation of a Random Forest Machine Learning Algorithm, University of Manchester, Manchester, UK, 2012.
[34] A.R. Webb, Ensemble methods, in: Statistical Pattern Recognition, John Wiley & Sons, United Kingdom, 2011, pp. 361–403.
[35] T. Lindgren, Random rule sets–combining random covering with the random subspace method, Int. J. Mach. Learn. Comput. 8 (1) (2018) 8–13.
[36] S. Džeroski, P. Panov, B. Ženko, Machine learning, ensemble methods in, in: Computational Complexity: Theory, Techniques, and Applications, Springer New York, New York, NY, 2012, pp. 1781–1789.
[37] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (7) (1997) 1145–1159.
[38] I. Markoulidakis, I. Rallis, I. Georgoulas, G. Kopsiaftis, A. Doulamis, N. Doulamis, Multiclass confusion matrix reduction method and its application on net promoter score classification problem, Technologies 9 (4) (2021) 81.