Overground vs. treadmill: where is my horse running? Context detection based on IMU data

ZIHAO XU, Electrical Engineering, Mathematics and Computer Science (EEMCS) University of Twente, The Netherlands

Horse gaits and lameness can be classified and studied either in-hand overground or on a treadmill, using, for example, optical motion capture (OMC) or inertial measurement unit (IMU) systems. Therefore, the context of horse locomotion bears particular importance and value in the field of research. Especially, treadmill evaluations enable gathering highly standardized data at a steady state. Research on developing classification models for overground vs. treadmill locomotion is solicited to improve the automation of an existing gait analysis system. This research will train respectively three feature-based machine learning (ML) and two signal-based deep learning (DL) models by the collected data, with IMUs mounted on both upper-body (poll, withers, pelvis, see in Figure 1) and limbs, from overground and treadmill terrain types. The data are then divided into three datasets, followed by an analysis evaluating different approaches' performances on each set. This research reveals that ML models generally achieve considerably higher accuracy and stability than DL models. Besides, ML models applied with the feature reduction technique experienced a performance drop.

Additional Key Words and Phrases: terrain classification, inertial measurement unit, machine learning, deep learning, feature extraction

1 INTRODUCTION

Horse locomotion reflects orthopedic status condition. Disorder of the locomotor system, clinically manifested as lameness, is one of the main reasons for veterinary consultation [20]. Lameness has induced significant financial loss each year [25, 32]. However, when diagnosing lameness, even an expert's judgment could suffer from the limitation of human visual perception of asymmetry [21].

To address these problems, researchers have proposed different models using data gathered by different hardware [27, 28, 34]. These models have their respective strengths and limitations, discussed in corresponding papers. However, IMU is considered to be a cost-effective choice [3, 19]. As a result, a system called "EquiMoves" was developed, to objectively examine horse gaits, utilizing kinematics parameters by ML or DL to analyze data collected from IMU [3]. The developed system could also aid veterinarians during gait analysis, which can detect gait changes due to lameness or performance degradation, with a good level of accuracy.

Terrain type would influence the current system accuracy since the trained model was developed with IMU data from the overground surface. Horses running on a treadmill or overground have different kinematic features, resulting in dissimilar IMU signals; for example, hind limbs are placed earlier than the forelimbs in making diagonal ground contact in overground locomotion, whereas the stance duration of the forelimbs is more prolonged than either overground condition in treadmill locomotion [6]. The existing system's analysis performances would be undermined, thus requiring a precise classification for the collected data to extrapolate where the horse is running for better automation. Besides, most of the gait events algorithms were developed overground [4]; transferring these algorithms to treadmill locomotion can lead to worse accuracy.

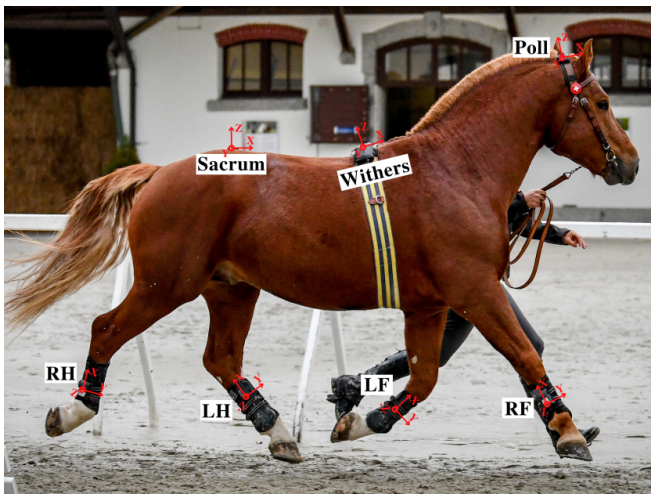

Fig. 1. IMUs locations and orientations on horse body. RF: right front limb, LF: left front limb, RH: right hindlimb, LH: left hindlimb. Photographed by Christelle Althaus. [8]

In this research, Data from twelve horses of two breeds: (1) 7 trotters and (2) 5 warmbloods, on both treadmill and overground, will be used, which was collected with body and limb mounted IMUs, see in Figure 1. A detailed description of dataset division can be found in Section 5.1. After extracting features, ML performances with and without feature reduction technique would be investigated. This research would mainly focus on comparing differences in the results obtained from ML and DL algorithms and determining which type achieves the overall best accuracy and stabilitiy. The hypothesis is that DL models would outperform ML models in the context of terrain classification with IMU sensors.

2 PROBLEM STATEMENT

There have been many works concerning horses' gait analysis and detection of lameness [3, 26–28, 34]; however, less has been done for classifying the type of terrain on which the horse is running. This classification is of importance to automate the gait analysis process. Because of that, this paper will utilize previously collected IMUs data to develop several classification models.

3 RESEARCH QUESTION

The main research question would be which type of model, either the feature-based ML model, such as a Support-vector machine (SVM), or the signal-based DL model, for example, Recurrent neural

TScIT 37, July 8, 2022, Enschede, The Netherlands

network (RNN), gives the most accurate prediction for overground vs. treadmill locomotion classification. The main research question can further be divided into three sub-questions.

3.1 Sub-Question 1

Which ML models and DL models would be selected for classification?

3.2 Sub-Question 2

How to select features, and whether feature reduction would yield better results?

3.3 Sub-Question 3

which one of the chosen ML and DL models has the respective best performance?

4 RELATED WORK

To conduct the research, previous research related to this field should be studied. Scopus and Google scholar are chosen as media to look up information. Keywords such as "Terrain types AND IMU," "Treadmill and overground," and "terrain classification AND IMU," enable to find the following work.

In 1994, the values of comparing treadmill kinematic data and overground ones had already merited researchers' attention. A study was done by using the optoelectronic CODA-3 analysis system to gather kinematics data of horses on a rubber floor, asphalt floor (overground), and a treadmill, finding out many differences when the terrain types differ, and the transfer of data in different domains should be attended to with special care [6].

In 2008, a neural network classification model, utilizing IMU of three axes acceleration and three axes angular velocity data, was developed intended for the autonomous vehicle to classify driving terrain types into five categorizations: flat plane, rugged terrain, grassy terrain, incline plane and unclassified, yielding very accurate results [15].

In 2019, two studies were presented to study human gait, and conduct a classification of locomotion context [9, 13]. Both studies collected data through IMU sensors, with differences that Hashimi et al. (2019)'s research is based on human walking data and that classification models are traditional feature-based algorithms; however, Dixon et al. (2019)'s research is performed on human running data, and that classification models were a comparison of one feature-extracted model, Gradient Boosting(GB) and one signal-based model Convolutional neural network (CNN). Nevertheless, Both have shown that the trained models demonstrate good abilities to classify terrain types with high probabilities above 90%.

5 METHODOLOGIES

Herein an approach to answering this paper's research questions will be detailed. The respective performances of feature-based ML models and signal-based DL models must be evaluated first. Subsequently, a comparison between ML and DL models will answer the main research question. Data processing and results analyses were conducted in Matlab 2022a [18].

Table 1. Raw datasets extraction conditions

Subset Index	Horse Type	Gait Label	Movement Direction	Sensors
1	Trotters & Warmblood	Walk	Straight	low and high -g accelerometers & gyroscopes
2	Warmblood	Walk& Trot	Straight	low and high -g accelerometers & gyroscopes
3	Trotters& Warmblood	Walk& Trot	Straight	low and high -g accelerometers & gyroscopes

5.1 Data Collection

Data was collected using a wireless network of IMUs sensors mounted on the poll, withers, sacrum, and each limb parts of horses both overground and on the treadmill. For detailed sensor types and parameters, see in [7] (ProMove Mini, Inertia Technology B.V., Enschede, The Netherlands). Specifically, data from two types of accelerometers and gyroscopes, which have a sampling frequency of 200 Hz, are recorded in three dimensions, x,y, and z. The corresponding results, including terrain types, gait classes, horse directions, start and stop of strides, and other locomotion analysis parameters, were automated and labeled based on the paper from Serra Braganca et al. (2020) [26]. Only low -g, high -g accelerometers and gyroscopes data from each limb, wither, and sacrum was utilized for this study. Besides, since horse locomotion systems would differ in various breeds [24], thus presenting various motion patterns, trotters, and warmblood breeds for a more generalizable outcome were investigated [8]. Given that treadmill data is always being labeled as straight, only horses marked with straight were incorporated into the new raw dataset of interest. Furthermore, since the available data applied with previous conditions results that trotters have both walk and trot gaits data available overground, but only walk type data was gathered on the treadmill, whereas warmblood horses have complete types, this induced a sub-division of filtered data into three subsets: (1) Gait type walk data only using both trotters and warmblood, (2) Gait type trot data only using only warmblood, and (3) data having everything pooled together. See in Table 1. A detailed discussion with three subsets is mentioned in Section 5.3. Besides, since data labeled as either walk or trot were recorded alternatively, the retrieved dataset was thus also following this pattern. Finally, These altogether lead to a 186 entries dataset(without sub-division yet), each of which has its corresponding series length (length of time stamps) x 9 (tri-axial low g, high g accelerometers, and gyroscopes) of the respective five above-mentioned body parts.

5.2 Data Pre-Processing

Data should be augmented to have clearer spectral characteristics. Therefore, a sliding window of a fixed size of 200 was implemented, which counts up to 1 second since the sampling rate is 200Hz. After applying this algorithm, the previously retrieved 186 entries were

Table 2. Composition of subsets labels

Subset Index	Overground Labels	Treadmill Labels	Total
1	3086	4343	7429
2	623	906	1529
3	4670	5249	9919

further expanded to 9919 records, with each entry having 200(fixed window size) x 9 (tri-axial of three types of sensors) data of six body parts. Besides, each of these nine sensors data was normalized by min-max normalization, with minimum and maximum values decided by each of 9919 records, respectively. (Formula 1 [22]). Before extracting features for ML models, since the collected data is signal-based, it is, therefore, necessary to window data to reduce spectral leakage. Hann window, see in Formula 2, generally works well in this context [5]. An overlapping rate of 50% was selected for optimal performance [12].

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

5.3 Feature Extraction

According to Darbandi et al., spatial domain features, such as minimum, maximum, and Kurtosis, and frequency domain features, such as Fourier transform coefficients, phase angles, and spectral energy were extracted [8]. The algorithm adopted to obtain spectral pattern characteristics was Fast Fourier Transformation (FFT). A total number of 25 features were calculated per signal, sensor and node. Then, in the 9919 records, each entry comprises 25 features in 54 (9 sensors x 6 body parts) x 1 format. Since this whole process was performed on the entire Trotters and Warmblood breeds datasets, the sub-divisions conditions mentioned in section 5.1 should be applied. The composition in terms of overground and treadmill labels of final subsets are in Table 2. All subsets have balanced labels, so no further up-sampling or down-sampling sample process actions are needed.

$$w(n) = 0.5(1 - cos(2\pi\frac{n}{N})), 0 \leq n \leq N \tag{2}$$

5.4 Models Selection

5.4.1 Machine Learning. The three subsets are relatively small in sample size but high in dimensions: 54(9 sensors of 6 body parts) x 25(features) = 1350(a total number of features per sensor and node), and the classification task is a binary classification. Therefore, SVM [11], Logistic Regression (LR) [1], and for its simplicity [23], but also a good performance on classification, K-nearest neighbors algorithm (KNN) were also chosen for evaluation.

5.4.2 Deep Learning. Long short-term memory (LSTM) usually obtains good results when solving sequential data [10]. However, LSTM is more computationally costly than CNN. Dixon et al.(2019) also achieved similar performances in other terrain classification research with CNN [9]. Therefore, CNN is the first choice for DL Model. To compensate for LSTM's cost in time consumption, a CNN-LSTM model is selected, which not only exploits the benefits of having CNN extracting features, naturally, reducing feature sizes but also has the effectiveness of dealing with time-series data as LSTM [17]. Therefore, only CNN and CNN-LSTM models were determined to be trained.

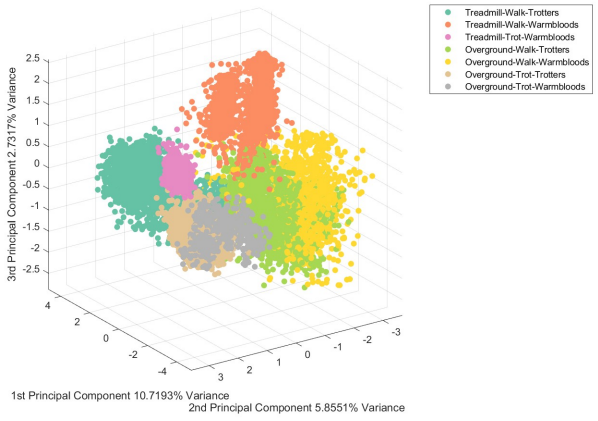

Fig. 2. Spaces of First Three Principal Components on Dataset 3 (Treadmill-Trot-Trotters data was not recorded.)

5.5 Experiment on Feature Dimensions Reduction

As 1350 features for ML are comparably high in dimensions, Principal component analysis (PCA) was exploited to reduce feature size. Normalization by minimum and maximum values of each feature was applied to the dataset in the first place. PCA achieves good reduction performance that dimensions were decreased from 1350 to 376 with features contribution threshold set to 0.95. A Visualization of data representation in the first three principal components spaces is shown in Figure 2. The corresponding most important features of the three components can be found in Table 3. Dataset 3 was chosen for illustrating since all data were pooled together for a complete view.

5.6 Experiment On Machine Learning

5.6.1 SVM. Given high dimensional data characteristics and classification of two classes, a linear kernel, hinge loss, and lasso regularization [2, 16], were picked for the model. A five-fold cross-validation model was developed to find the best Lambda by the point which predictor variable sparsity and classification error are balanced, according to [31], see Figure 3 for an example. This process was repeated for all three datasets. Models' parameters derived this way were then stored individually for later training, during which, because of relatively smallness in a total number of horses and the attempt to preclude trained horses from reappearing in testing data, the Leave-One-Out (LOO) method was taken in a way that each horse was treated as testing data once, and the final accuracy was calculated based on the mean of them. This LOO approach would be adopted in other ML models.

Table 3. Most Important Features In Spaces Of The First Three Principal Components Based On Dataset 3.

Principal Component Index	Feature Name	Z-score Normalization Value Of Each Principal Component (mean 0 and standard deviation 1)
1	poll_Min_lg_z	3.582107651461500
2	lh_Median_lg_x	4.004266546028038
3	rf_Median_lg_z	4.096538684160307
poll_Min_lg_z: The Min value calculated by z axis data of low-g accelerometers from the poll of horses. lh_Median_lg_x: The median value calculated by x axis data of low-g accelerometers from the left hind limb of horses. rf_Median_lg_z: The median value calculated by z axis data of low-g accelerometers from the right front limb of horses.		

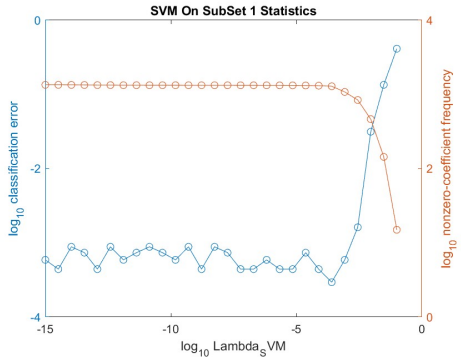

Fig. 3. Classification Loss By Cross Validation Tested On Dataset 1. Axis nonzero coefficient frequency stands for regularization strength. Index 11 is chosen in this example with lambda value set to 4.641588833612792e-11.

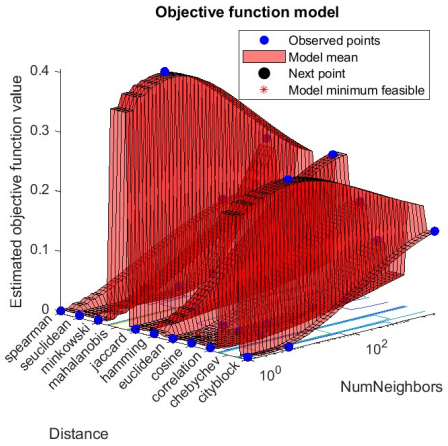

Fig. 4. Best Observed Points On Dataset 1 By KNN Hyperparameters Optimization. Distance description can be found in [30].

5.6.2 Logistic Regression. Solver type Sparse is widely used when dealing with high dimensional data [1]. Apart from this difference, the logistic regression model and the corresponding accuracy were investigated similarly to those of SVM.

5.6.3 KNN. KNN models only require neighbor numbers and distance type two parameters, which are tuned and optimized by different combinations. Best observed feasible points were explored with the help of Matlab library [30], see Figure 4, and their parameters on three datasets were stored for training.

5.7 Experiment On Deep Learning

5.7.1 CNN. CNN models were tuned based on examples provided by Matlab documentation [29]. Some examples are given with a similar "time-series sequence to label solving" approach. Separate datasets adjusted the models in this research. Besides, data were expanded from the original entry number x 200 (window size) * 54 (feature sizes) to 200 * 54 * 1 * entry number before feeding into CNN. The models start with a 200 x 54 single-channel input layer with zero center normalization. Three 2D convolutional layers were used, with respective filter numbers 6, 12, and 24 of size 2 x 4. Each layer was further linked by a height one and width two max-pooling layer with a vertical step of 1 and horizontal step of 2, followed by a ReLu layer. The dropout layer was applied before a fully connected layer with two outputs connected to a classification layer, as shown in Figure 5 for an example. The learning rate was experimented with 0.002, with a max of 5 epochs and 50,100,200 mini-batch sizes. Furthermore, the LOO method was exploited in both CNN and CNN-LSTM.

```
layers =

  13×1 Layer array with layers:

     1   'input'   Image Input             200×54×1 images with 'zerocenter' normalization
     2   'conv1'   Convolution             6 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     3   'relu1'   ReLU                    ReLU
     4   'pool1'   Max Pooling             1×2 max pooling with stride [1  2] and padding [0  0  0  0]
     5   'conv2'   Convolution             12 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     6   'relu2'   ReLU                    ReLU
     7   'pool2'   Max Pooling             1×2 max pooling with stride [1  2] and padding [0  0  0  0]
     8   'conv3'   Convolution             24 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     9   'relu3'   ReLU                    ReLU
    10   ''        Dropout                 50% dropout
    11   ''        Fully Connected         2 fully connected layer
    12   ''        Softmax                 softmax
    13   ''        Classification Output   crossentropyex
```

Fig. 5. CNN architecture on dataset 1

5.7.2 CNN-LSTM. CNN-LSTM models originated from the same documentation as well as that of CNN [29]. Data with format entry number x 200 x 54 was directly fed into the sequence input layer without additional processing. Following the sequence input layer, a folding layer was utilized to convert sequence data to images like structures for later convolution. A Relu, dropout, batch normalization, and average pooling layer are subsequent after a 2D convolutional layer. Finally, data was unfolded and passed into an LSTM layer. Learning rate, batch sizes, and others were the same as CNN's. A detailed structure and parameters can be found in Figure 8.

```
layers =

  15×1 Layer array with layers:

     1   'input'   Image Input             200×54×1 images with 'zerocenter' normalization
     2   'conv1'   Convolution             6 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     3   'relu1'   ReLU                    ReLU
     4   ''        Dropout                 50% dropout
     5   'pool1'   Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
     6   'conv2'   Convolution             12 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     7   'relu2'   ReLU                    ReLU
     8   ''        Dropout                 50% dropout
     9   'pool2'   Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
    10   'conv3'   Convolution             24 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
    11   'relu3'   ReLU                    ReLU
    12   ''        Dropout                 50% dropout
    13   ''        Fully Connected         2 fully connected layer
    14   ''        Softmax                 softmax
    15   ''        Classification Output   crossentropyex
```

Fig. 6. CNN architecture on dataset 2

```
layers =

  15×1 Layer array with layers:

     1   'input'   Image Input             200×54×1 images with 'zerocenter' normalization
     2   'conv1'   Convolution             8 3×5 convolutions with stride [1  1] and padding [0  0  0  0]
     3   'relu1'   ReLU                    ReLU
     4   ''        Dropout                 50% dropout
     5   'pool1'   Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
     6   'conv2'   Convolution             16 3×5 convolutions with stride [1  1] and padding [0  0  0  0]
     7   'relu2'   ReLU                    ReLU
     8   ''        Dropout                 50% dropout
     9   'pool2'   Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
    10   'conv3'   Convolution             32 3×5 convolutions with stride [1  1] and padding [0  0  0  0]
    11   'relu3'   ReLU                    ReLU
    12   ''        Dropout                 50% dropout
    13   ''        Fully Connected         2 fully connected layer
    14   ''        Softmax                 softmax
    15   ''        Classification Output   crossentropyex
```

Fig. 7. CNN architecture on dataset 3

```
21×1 Layer array with layers:

     1   'input'      Sequence Input          Sequence input with 200×54×1 dimensions
     2   'fold'       Sequence Folding        Sequence folding
     3   'conv1'      Convolution             2 4×6 convolutions with stride [1  1] and padding [0  0  0  0]
     4   'relu1'      ReLU                    ReLU
     5   'drop1'      Dropout                 50% dropout
     6   ''           Batch Normalization     Batch normalization
     7   'pool1'      Average Pooling         2×4 average pooling with stride [1  2] and padding [0  0  0  0]
     8   'conv2'      Convolution             4 4×6 convolutions with stride [1  1] and padding [0  0  0  0]
     9   'relu2'      ReLU                    ReLU
    10   'drop2'      Dropout                 50% dropout
    11   ''           Batch Normalization     Batch normalization
    12   'pool2'      Average Pooling         2×4 average pooling with stride [1  2] and padding [0  0  0  0]
    13   'conv3'      Convolution             8 4×6 convolutions with stride [1  1] and padding [0  0  0  0]
    14   'relu3'      ReLU                    ReLU
    15   'unfold'     Sequence Unfolding      Sequence unfolding
    16   'flatten1'   Flatten                 Flatten
    17   'lstm'       LSTM                    LSTM with 7 hidden units
    18   'drop3'      Dropout                 50% dropout
    19   'fc'         Fully Connected         2 fully connected layer
    20   'softmax'    Softmax                 softmax
    21   'output'     Classification Output   crossentropyex
```

Fig. 8. CNN-LSTM architecture on dataset 1

```
17×1 Layer array with layers:

     1   'input'      Sequence Input          Sequence input with 200×54×1 dimensions
     2   'fold'       Sequence Folding        Sequence folding
     3   'conv1'      Convolution             10 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     4   'relu1'      ReLU                    ReLU
     5   'pool1'      Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
     6   'conv2'      Convolution             20 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     7   'relu2'      ReLU                    ReLU
     8   'pool2'      Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
     9   'conv3'      Convolution             40 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
    10   'relu3'      ReLU                    ReLU
    11   'unfold'     Sequence Unfolding      Sequence unfolding
    12   'flatten1'   Flatten                 Flatten
    13   'lstm'       LSTM                    LSTM with 5 hidden units
    14   'drop1'      Dropout                 50% dropout
    15   'fc'         Fully Connected         2 fully connected layer
    16   'softmax'    Softmax                 softmax
    17   'output'     Classification Output   crossentropyex
```

Fig. 9. CNN-LSTM architecture on dataset 2

6 RESULTS

Treadmill and Overground classification by ML and DL algorithms over three datasets results are presented in this section.

```
17×1 Layer array with layers:

     1   'input'      Sequence Input          Sequence input with 200×54×1 dimensions
     2   'fold'       Sequence Folding        Sequence folding
     3   'conv1'      Convolution             7 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     4   'relu1'      ReLU                    ReLU
     5   'pool1'      Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
     6   'conv2'      Convolution             14 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
     7   'relu2'      ReLU                    ReLU
     8   'pool2'      Average Pooling         1×2 average pooling with stride [1  2] and padding [0  0  0  0]
     9   'conv3'      Convolution             28 2×4 convolutions with stride [1  1] and padding [0  0  0  0]
    10   'relu3'      ReLU                    ReLU
    11   'unfold'     Sequence Unfolding      Sequence unfolding
    12   'flatten1'   Flatten                 Flatten
    13   'lstm'       LSTM                    LSTM with 14 hidden units
    14   'drop1'      Dropout                 50% dropout
    15   'fc'         Fully Connected         2 fully connected layer
    16   'softmax'    Softmax                 softmax
    17   'output'     Classification Output   crossentropyex
```

Fig. 10. CNN-LSTM architecture on dataset 3

6.1 Machine Learning

ML generally revealed good results on all datasets around 95%, in which SVM and KNN both attain good accuracy; however, KNN turns out to be more stable when all data pooled together, as shown in Table 4. The corresponding confusion matrixes are represented in Appendix A, calculated by concatenating all prediction results of each LOO against true labels.

6.2 Feature Reduction

With the contribution threshold set to 0.95, the final mean results of all different ML models suffer from an significant accuracy drop, as shown in Table 5.

6.3 Deep Learning

DL models only revealed good results on dataset two, around 98%, but around 80 % on other datasets, as shown in Table 6. NaN value was observed when CNN was tested on trotter 5 from dataset 3, and this was removed when calculating the mean and standard deviation. The corresponding confusion matrixes are shown in Appendix B, and calculation approach is the same as that of ML.

7 DISCUSSION

Surprisingly, experimental results contradict the start-off assumption that DL algorithms would outperform ML algorithms. The ML algorithms generally achieved good performance, whereas either DL outperformed or no significant difference observed was reported in related studies [9, 33]. A closer look at data identifies that these ML algorithms obtain comparably higher results on dataset 1. The possible explanation was that using walk data from both breeds precludes influences incurred by different gait types, as the case in dataset 3, and higher than dataset 2 could be accounted as relative small sizes, which is four times smaller than that of dataset 1, and models, therefore, lack of sufficient training. Among three ML algorithms, it can be seen that KNN is the model having average sound performance and slight variances on different tested horses. KNN hence appears to be a general reliable algorithm in this study.

As for PCA, The upsides are owing to decreasing dimensions, less computation time, yet memory load was experienced. However, considering relatively acceptable training time with and without feature reduction, these advantages do not outweigh the improvements brought by using all features. Therefore, it is suggested to

Table 4. Average performances of different ML models on three datasets.

Dataset Index	LR				SVM				KNN			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
1	97.22(7.06)	96.83(8.65)	98.44(2.06)	97.46(5.43)	**98.77(2.85)**	98.14(5.43)	99.49(1.22)	98.72(2.99)	97.88(2.20)	97.61(4.28)	97.72(2.24)	97.60(2.40)
2	94.10(11.35)	99.94(00.19)	93.54(17.21)	95.62(12.38)	**97.87(3.60)**	97.14(9.46)	99.13(1.35)	97.92(5.97)	93.49(7.43)	99.68(1.12)	92.04(12.19)	95.27(7.34)
3	95.35(11.08)	94.30(13.25)	98.99(1.36)	96.10(8.29)	93.75(11.74)	92.07(14.93)	99.62(0.64)	95.04(9.30)	**96.18(2.44)**	97.35(5.49)	94.55(3.52)	95.80(2.95)

Units: Mean (standard deviation) % calculated by three ML algorithms with data collected from each limb, sacrum and wither on three sub-datasets.
Precision, Recall, F1 were calculated on Treadmill label. The highest accuracy values on each set are shown in bold.

Table 5. Average performances of different ML models with feature reduction technique on three datasets.

Dataset Index	LR				SVM				KNN			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
1	86.20(11.67)	87.40(18.58)	85.86(11.83)	84.76(11.90)	**87.01(2.85)**	86.68(17.99)	89.76(10.81)	86.66(11.57)	76.91(14.57)	76.30(23.16)	79.52(13.14)	74.72(15.81)
2	83.20(16.60)	90.40(20.62)	83.40(12.43)	84.38(15.33)	83.71(3.60)	89.30(17.24)	88.19(11.21)	87.23(10.85)	**89.73(10.15)**	93.46(7.68)	75.97(17.74)	82.53(12.02)
3	**87.95(10.43)**	89.58(14.01)	88.36(9.46)	88.07(8.60)	87.85(11.74)	89.11(13.58)	89.79(10.18)	88.65(9.76)	80.46(11.24)	80.31(21.12)	85.12(9.23)	80.42(12.23)

Units: Mean (standard deviation) % calculated by three ML algorithms with data collected from each limb, sacrum and wither on three sub-datasets.
Precision, Recall, F1 were calculated on Treadmill label. The highest accuracy values on each set are shown in bold.

Table 6. Average performances of different DL models on three datasets.

Dataset Index	CNN				CNN-LSTM			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
1	80.51(26.45)	92.69(13.35)	78.62(39.31)	76.46(35.72)	**80.78(27.35)**	84.94(23.83)	79.75(35.87)	79.03(31.81)
2	96.24(4.43)	93.39(11.25)	99.38(1.79)	96.20(6.71)	**99.37(1.00)**	94.47(11.57)	99.24(2.21)	96.39(6.74)
3	**86.41(24.08)**	96.07(8.37)	84.14(33.53)	84.25(29.44)	83.17(11.74)	94.98(8.28)	72.28(39.77)	75.12(34.47)

Units: Mean (standard deviation) % calculated by three DL algorithms with data collected from each limb, sacrum and wither on three sub-datasets.
Precision, Recall, F1 were calculated on Treadmill label. The highest accuracy values on each set are shown in bold. **During LOO, CNN model misclassified all trotter 5 labels in dataset 3 and NaN value is therefore removed.**

utilize all features, which is congruent with findings by Dixon (2019) et al. [9]

DL experienced a notable performance drop-down when trained on dataset 1 and dataset 3, in which considerable standard deviation values indicate instability and overfitting problems were occurring in training. In contrast, dataset 2 reached an impressive accuracy with a much lower deviation, which could be explained by the biomechanical difference in horses' locomotion patterns [24] influencing data patterns. To further confirm this explanation, dataset 3 was accordingly adjusted to only classify trotters, and around 99% accuracy was then observed. This reflects DL's susceptibility. Interestingly, unlike ML algorithms had better performance on dataset 1 than 3, DL algorithms achieved better results on dataset 3 than 1. One interpretation for this would be that relatively larger sample sizes, 9919 versus 7429, enable better learning, and naturally, fewer deviations are observed since models learn more about data. Overall, CNN-LSTM yielded a better outcome than CNN, considering both accuracy and stability.

NaN data of CNN mentioned earlier was caused by all treadmill data being misclassified to overground labels on one testing horse, which is not observed in other models. However, this tendency to misclassify treadmill to overground is further observed on both ML, except for KNN, and DL models.

8 CONCLUSION

Many pieces of research have been done to analyze horse gaits [3, 3, 26, 28, 34]. However, little has been done concerning terrain type classification using IMU for horses, even though the difference of terrain types would result in non-trivial influences on kinematics data [6], and therefore affect the accuracy of these gaits analyses. Most existing terrain classification researches focus on Autonomous off-road driving and Human Gait [9, 13–15]. This research shows that ML and DL algorithms can classify terrain types using various sensors mounted on horses but also presents useful results and noteworthy phenomena in results, such as the NaN value, performance drop with feature reduction technique, etc. The findings would give insight into the context-awareness classification of horses overground or on a treadmill, and could be generalized to other domains following a similar method, and contribute to the automation of IMU-based gait analysis systems, such as the EquiMoves(R) [3].

9 FUTURE WORK

While having strength in this research, given time constraints and relatively small dataset sizes for DL, the most refined model parameters are not discovered in both ML and DL, typically in terms of DL. Besides, this research does not explore different sensor types and body parts combinations' influence on prediction performances. In addition, when breeds are mixed with different gaits, it seems that DL models would be subject to a negative influence on accuracy, as in the case of dataset 1 and dataset 3 versus dataset 2, and it would therefore merit further exploration. Future research could potentially shift focus to the aspects as mentioned above.

ACKNOWLEDGMENTS

Darbandi et al[8]. are acknowledged for the creation of the horse IMUs' locations graph (Figure 1).
The author would like to thank his supervisor, J.I.M. Parmentier, for her encouragement and support.

REFERENCES

[1] Felix Abramovich and Vadim Grinshtein. 2018. High-dimensional classification by sparse logistic regression. *IEEE Transactions on Information Theory* 65, 5 (2018),

3068–3079.
[2] Yi Bao, Tao Wang, and Guoyong Qiu. 2014. Research on applicability of SVM kernel functions used in binary classification. In *Proceedings of International Conference on Computer Science and Information Technology*. Springer, 833–844.
[3] Stephan Bosch, Filipe Serra Bragança, Mihai Marin-Perianu, Raluca Marin-Perianu, Berend Jan Van der Zwaag, John Voskamp, Willem Back, René Van Weeren, and Paul Havinga. 2018. EquiMoves: a wireless networked inertial measurement system for objective examination of horse gait. *Sensors* 18, 3 (2018), 850.
[4] FM Bragança, S Bosch, JP Voskamp, Mihai Marin-Perianu, BJ Van der Zwaag, JCM Vernooij, PR van Weeren, and Willem Back. 2017. Validation of distal limb mounted inertial measurement unit sensors for stride detection in Warmblood horses at walk and trot. *Equine veterinary journal* 49, 4 (2017), 545–551.
[5] S. Braun. 2001. WINDOWS. In *Encyclopedia of Vibration*, S. Braun (Ed.). Elsevier, Oxford, 1587–1595. https://doi.org/10.1006/rwvb.2001.0052
[6] HHF Buchner, HHCM Savelberg, HC Schamhardt, HW Merkens, and A Barneveld. 1994. Kinematics of treadmill versus overground locomotion in horses. *Veterinary Quarterly* 16, sup2 (1994), 87–90.
[7] Inertia Technology B.V. 2019. *Promove-Mini datasheet V16 - Inertia Technology*. Enschede, The Netherlands. https://inertia-technology.com/wp-content/uploads/2019/08/ProMove-mini-datasheet.pdf
[8] Hamed Darbandi, Filipe Serra Bragança, Berend Jan Van der Zwaag, John Voskamp, Annik Imogen Gmel, Eyrún Halla Haraldsdóttir, and Paul Havinga. 2021. Using Different Combinations of Body-Mounted IMU Sensors to Estimate Speed of Horses—A Machine Learning Approach. *Sensors* 21, 3 (2021), 798.
[9] PC Dixon, KH Schütte, B Vanwanseele, JV Jacobs, JT Dennerlein, JM Schiffman, PA Fournier, and B Hu. 2019. Machine learning algorithms can classify outdoor terrain types during running using accelerometry data. *Gait & posture* 74 (2019), 176–181.
[10] Felix A Gers, Douglas Eck, and Jürgen Schmidhuber. 2002. Applying LSTM to time series predictable through time-window approaches. In *Neural Nets WIRN Vietri-01*. Springer, 193–200.
[11] Bissan Ghaddar and Joe Naoum-Sawaya. 2018. High dimensional data classification and feature selection using support vector machines. *European Journal of Operational Research* 265, 3 (2018), 993–1004.
[12] Fredric J Harris. 1978. On the use of windows for harmonic analysis with the discrete Fourier transform. *Proc. IEEE* 66, 1 (1978), 51–83.
[13] Muhammad Zeeshan Ul Hasnain Hashmi, Qaiser Riaz, Mehdi Hussain, and Muhammad Shahzad. 2019. What lies beneath one's feet? terrain classification using inertial data of human walk. *Applied Sciences* 9, 15 (2019), 3099.
[14] Boyi Hu, PC Dixon, JV Jacobs, JT Dennerlein, and JM Schiffman. 2018. Machine learning algorithms based on signals from a single wearable inertial sensor can detect surface-and age-related differences in walking. *Journal of biomechanics* 71 (2018), 37–42.
[15] Rubkwan Jitpakdee and Thavida Maneewarn. 2008. Neural networks terrain classification using inertial measurement unit for an autonomous vehicle. In *2008 SICE Annual Conference*. IEEE, 554–558.
[16] Hyungwoo Kim, Insuk Sohn, and Seung Jun Shin. 2021. Regularization paths of L 1-penalized ROC Curve-Optimizing Support Vector Machines. *Stat* 10, 1 (2021), e400.
[17] Ioannis E Livieris, Emmanuel Pintelas, and Panagiotis Pintelas. 2020. A CNN–LSTM model for gold price time-series forecasting. *Neural computing and applications* 32, 23 (2020), 17351–17360.
[18] MATLAB. 2022. *version 9.12.0 (R2022a)*. The MathWorks Inc., Natick, Massachusetts.
[19] MJ McCracken, J Kramer, KG Keegan, M Lopes, DA Wilson, SK Reed, A LaCarrubba, and M Rasch. 2012. Comparison of an inertial sensor system of lameness quantification with subjective lameness evaluation. *Equine Veterinary Journal* 44, 6 (2012), 652–656.
[20] T. D. Nielsen, R. S. Dean, N. J. Robinson, A. Massey, and M. L. Brennan. 2014. Survey of the UK veterinary profession: Common species and conditions nominated by veterinarians in practice. *Veterinary Record* 174, 13 (2014), 324–324. https://doi.org/10.1136/vr.101745
[21] RSV Parkes, R Weller, AM Groth, S May, and T Pfau. 2009. Evidence of the development of 'domain-restricted'expertise in the recognition of asymmetric motion characteristics of hindlimb lameness in the horse. *Equine Veterinary Journal* 41, 2 (2009), 112–117.
[22] S Patro and Kishore Kumar Sahu. 2015. Normalization: A preprocessing stage. *arXiv preprint arXiv:1503.06462* (2015).
[23] Hadi Raeisi Shahraki, Saeedeh Pourahmad, and Najaf Zare. 2017. Important neighbors: A novel approach to binary classification in high dimensional data. *BioMed research international* 2017 (2017).
[24] Justine J Robilliard, Thilo Pfau, and Alan M Wilson. 2007. Gait characterisation and classification in horses. *Journal of Experimental Biology* 210, 2 (2007), 187–197.
[25] Ann Hillberg Seitzinger, JL Traub-Dargatz, AJ Kane, CA Kopral, PS Morley, LP Garber, WC Losinger, and GW Hill. 2000. A comparison of the economic costs of equine lameness, colic, and equine protozoal myeloencephalitis (EPM). In *Proceedings of the 9th International Symposium on Veterinary Epidemiology and Economics*. 1–3.
[26] FM Serra Braganca, S Broomé, Marie Rhodin, S Björnsdóttir, V Gunnarsson, JP Voskamp, E Persson-Sjodin, W Back, Gabriella Lindgren, M Novoa-Bravo, et al. 2020. Improving gait classification in horses by using inertial measurement unit (IMU) generated data and machine learning. *Scientific reports* 10, 1 (2020), 1–9.
[27] Filipe M Serra Bragança, Elin Hernlund, Maj H Thomsen, Nina M Waldern, Marie Rhodin, Anna Byström, P René van Weeren, and Michael A Weishaupt. 2021. Adaptation strategies of horses with induced forelimb lameness walking on a treadmill. *Equine veterinary journal* 53, 3 (2021), 600–611.
[28] Aman Shrestha, Julien Le Kernec, Francesco Fioranelli, John F Marshall, and Lance Voute. 2017. Gait analysis of horses for lameness detection with radar sensors. (2017).
[29] Inc. The MathWorks. 2022. *Deep Learning with Time Series and Sequence Data*. Natick, Massachusetts, United State. DeepLearningwithTimeSeriesandSequenceData
[30] Inc. The MathWorks. 2022. *fitcknn*. Natick, Massachusetts, United State. https://nl.mathworks.com/help/stats/fitcknn.html
[31] Inc. The MathWorks. 2022. *fitclinear*. Natick, Massachusetts, United State. https://nl.mathworks.com/help/stats/fitclinear.html?searchHighlight=fitclinear&s_tid=srchtitle_fitclinear_1
[32] USDA. 2001. National economic cost of equine lameness, colic, and equine protozoal myeloencephalitis in the United States. *Information sheet* (2001).
[33] Fabio Vulpi, Annalisa Milella, Roberto Marani, and Giulio Reina. 2021. Recurrent and convolutional neural networks for deep terrain classification by autonomous robots. *Journal of Terramechanics* 96 (2021), 119–131.
[34] Tarik Yigit, Feng Han, Ellen Rankins, Jingang Yi, Kenneth McKeever, and Karyn Malinowski. 2020. Wearable IMU-based early limb lameness detection for horses using multi-layer classifiers. In *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*. IEEE, 955–960.

10 APPENDIX

A MACHINE LEARNING

A.1 LR

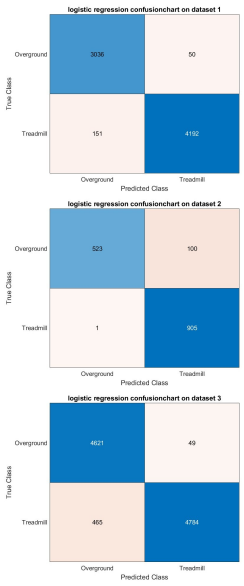

Fig. 11. Confusion Matrix of LR on three datasets (random seed fixed).

A.2 SVM

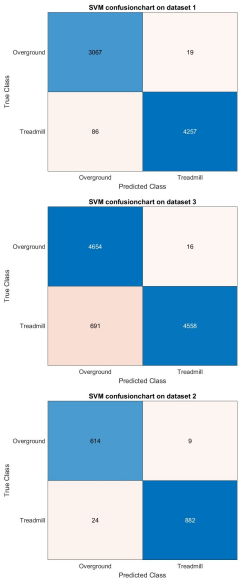

Fig. 12. Confusion Matrix of SVM on three datasets (random seed fixed).

A.3 KNN

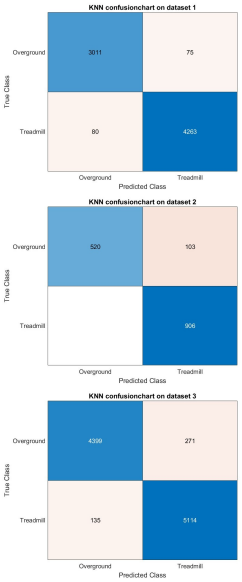

Fig. 13. Confusion Matrix of KNN on three datasets (random seed fixed).

B DEEP LEARNING

B.1 CNN

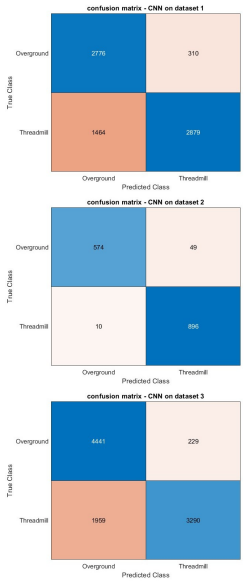

Fig. 14. Confusion Matrix of CNN on three datasets (random seed fixed).

B.2 CNN-LSTM

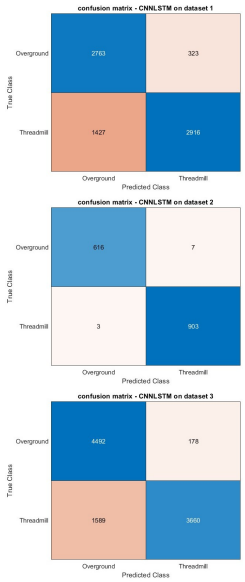

Fig. 15. Confusion Matrix of CNN-LSTM on three datasets (random seed fixed).