IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 6 March 2023, accepted 13 March 2023, date of publication 17 March 2023, date of current version 23 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3258621

RESEARCH ARTICLE

Semi-Supervised Machine Learning for Livestock Threat Classification Using GPS Data

URS J. DE SWARDT AND HERMAN KAMPER
Department of Electrical and Electronic Engineering, Stellenbosch University, Stellenbosch 7600, South Africa
Corresponding author: Urs J. de Swardt (21948828@sun.ac.za)

The work was financially supported by FarmRanger.

ABSTRACT South African livestock farmers face major challenges in the form of livestock theft and predation. In response to these concerns, farmers started using a collar that monitors the acceleration of an animal and, when specific parameters are met, triggers an alarm which transmits GPS data to the user's mobile application. Typically, a collar is placed on one animal per flock or herd. In this work, we aim to classify the GPS trajectories captured by these devices into four categories: theft, predation, own-handling and other. We lay particular emphasis on distinguishing theft alarms since these have direct implications for the safety and financial sustainability of farmers. To date, just over one million of these alarms have been recorded. Unfortunately, these trajectories are not labelled with the four categories. Therefore, we start by collecting labelled data sets that can be used for training classification models. We then investigate supervised and semi-supervised approaches for classifying the trajectories. Our semi-supervised approach shows the best results with performance comparable to human performance. The approach consists of three parts. First, an autoencoder and classifier are jointly trained to produce fixed-dimensional embeddings from GPS trajectories. Second, these embeddings are clustered to produce cluster labels. And lastly, the cluster labels are added to human-engineered features and used to train a final classifier. Our semi-supervised approach achieves an overall classification accuracy of 69%, with an F_1 score of 56% for theft events (4% lower than human performance) and an F_1 score of 90% for own-handling events (slightly outperforming a human). This model can be deployed to aid farmers in terms of safety and security by providing them with critical information in emergency situations.

INDEX TERMS Animal behavioral classification, GPS trajectory, IoT, machine learning.

I. INTRODUCTION

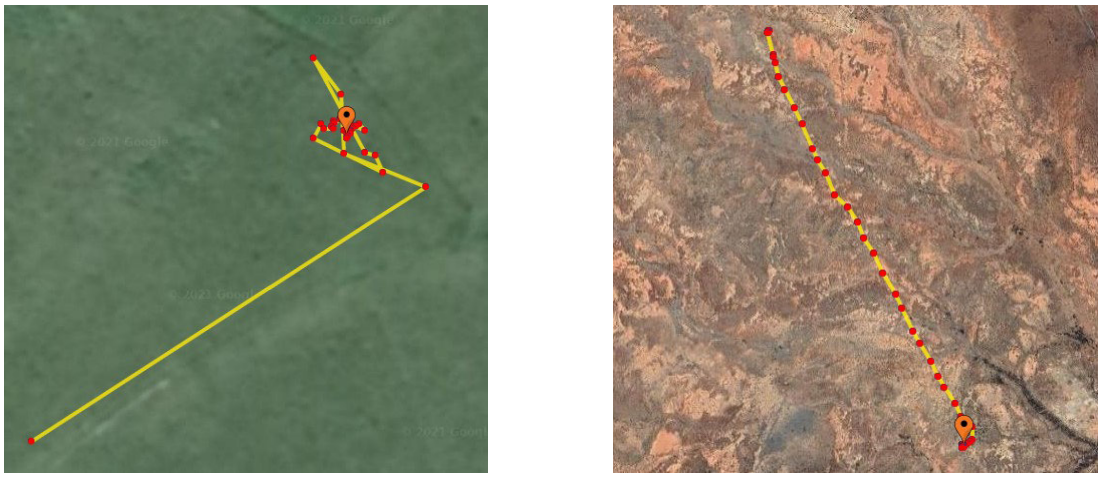
A. CONTEXT AND MOTIVATION

South Africa is experiencing high rates of farm murders [1] and livestock theft [2]. Livestock farmers also have to deal with the crippling cost of predator animals hunting livestock, estimated to amount to an annual loss of 13% for production animals [3]. In an attempt to alleviate these issues, FarmRanger developed an internet-of-things (IoT) device in 1999 that thousands of farmers now use to protect their livestock from theft and predation. A single device is attached to one animal per flock or herd. The unit monitors the animal's acceleration and when certain conditions are met,[1] it triggers an alarm that transmits GPS data to the owner's mobile application. To date, just over a million of these GPS trajectories have been recorded. However, it is unknown what happened during each of these events. It could have been theft, predation or one of several other possibilities that can cause rapid movement of the animals.

Figure 1 shows two examples of what a user would typically see on their mobile application. From these two examples, the reader can already imagine that a farmer would respond differently to each event depending on what is disturbing the animal. The implications on the safety of a farmer

The associate editor coordinating the review of this manuscript and approving it for publication was Ines Domingues.

[1]For intellectual property purposes, the exact details of the alarm trigger algorithm cannot be disclosed in this paper, but all the relevant details of the captured GPS trajectories can and are discussed in this work.



(a) Example of a jackal attack.

(b) Example of theft.

FIGURE 1. The GPS data points of two examples, one showing a predation event (a) and the other a theft event (b). In (a) there is seemingly random movement without a sense that the animal is moving in a certain direction. On the other hand, in the case of (b), deliberate movement in one direction can be seen. Note that these are carefully selected examples, and not necessarily representative of the rest of the data, i.e. in many cases, it is much more difficult to make an easy classification between predation and theft.

due to armed theft versus that of a sheep being attacked by a jackal are drastically different — the first being a life-threatening situation, while the latter mainly has financial implications. With this in mind, we ask the question: is it possible to utilise livestock movement data in order to distinguish theft, predation, own-handling, and other events[2] from one another? Doing so would equip a farmer with the necessary knowledge to properly prepare for an emergency and ultimately help keep farmers safe. Concretely, our goal is the four-class classification problem with a special emphasis on the theft class.

B. METHODOLOGY

We investigate a supervised and semi-supervised approach for classifying GPS trajectories. To evaluate these approaches, a small labelled validation data set is developed on which cross-validation classification performance is reported. Figure 2 shows a condensed overview of the approaches covered in this paper and how they relate to one another.

1) SUPERVISED LEARNING

At the bottom of Figure 2, we follow a standard supervised machine learning approach. The raw data is processed to create a human-engineered, fixed-dimensional feature vector for each trajectory. For the supervised approach, we train and evaluate different classifier models on this feature vector.

2) UNSUPERVISED LEARNING

In our previous work [4], we showed by means of clustering metrics that classes can be distinguished to some degree in a purely unsupervised approach as shown at the top of Figure 2. In short, the unsupervised model consists of an autoencoder which is trained to reconstruct its trajectory input with the aim of capturing valuable features in its fixed, low-dimensional latent embedding. This latent embedding can then be used to distinguish events from one another. This model forms the foundation for our semi-supervised approach.

[2]Other movement alarms can be events like playing, lightning strikes, etc.

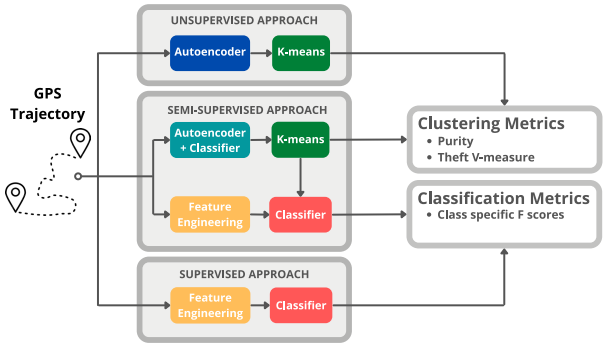


FIGURE 2. An overview of the methodology covered in this paper. Ultimately, we are only interested in classification performance metrics. However, since we use clustering as part of our semi-supervised approach, intermediate clustering metrics provide insights into how well we are extracting information from the abundant unlabelled data.

3) SEMI-SUPERVISED LEARNING

Our semi-supervised approach, as illustrated in the middle of Figure 2, extends the unsupervised model by jointly training a classifier alongside the autoencoder. The latent embedding is then clustered with K-means to produce a cluster label for each trajectory. Finally, the cluster labels are concatenated to the human-engineered feature vector used in the supervised approach before it is used to train classifier models.

4) EVALUATION

To evaluate the classification performance of the supervised and semi-supervised approaches, we perform cross-validation and report quantitative metrics. We find that the semi-supervised approach outperforms the supervised approach, especially in classifying theft events.

II. RELATED WORK

Literature directly related to our work is very limited and there are no other studies with which we can compare our performance directly. However, there are some similarities between our work, mode-of-transport classification, acoustic word embeddings and animal behaviour classification. Below we review these studies, but in every case also indicate how the data used in these types of studies differ from ours.

1) MODE-OF-TRANSPORT CLASSIFICATION

One other machine learning problem that also utilises GPS trajectories for classification, is the task of classifying mode-of-transport. This involves feeding GPS data points to a model that predicts whether a person is walking, driving, riding a bicycle etc. Various methods have achieved scores of up to 75% in classification accuracy [5]. One major difference in the context of mode-of-transport classification is that there is typically much more labelled data available. In the case of Microsoft's Geolife data set [6], there are 24 109 GPS trajectories labelled with a mode-of-transport and and 72 506 unlabelled trajectories. This is vastly more labelled data compared to our data set, which only contains a few hundred labelled trajectories. In addition, the time interval between data points is relatively small (1 to 5 seconds), in comparison to our data set (30 seconds). One approach proposed to classify mode-of-transport incorporates a convolutional-deconvolutional autoencoder to extract features from unlabelled data to assist in the supervised classification task [5]. In this model, an autoencoder and a classifier are trained jointly with weighted losses. We follow a similar route for our semi-supervised approach in Section VI.

2) ACOUSTIC WORD EMBEDDINGS

Our unsupervised trajectory embedding model discussed in Section I-B2 is heavily inspired by models from the area of speech processing, referred to as acoustic word embedding models [7], [8]. These models are similar to our model in the sense that they produce a fixed-dimensional representation of a time series — in the case of speech processing, the embedding is produced for a spoken utterance. The aim of these models is to produce embeddings where similar-sounding words are close to one another in the embedded space and dissimilar words are far from one another. Acoustic word embeddings precede the work in mode-of-transport classification mentioned above. Although we follow a similar methodology, the data used for acoustic word embeddings is fundamentally different from GPS trajectories in that the former involves acoustic time series.

3) ANIMAL BEHAVIOUR CLASSIFICATION

There are many studies related to animal behaviour classification. These studies use various sources of data in order to classify various animal activities. However, no studies attempt to classify threats to livestock animals directly. We elaborate on a few relevant studies here to contrast our work to existing research in the literature.

Le Roux et al. [9] used accelerometry data in a supervised setting to predict whether an animal is either lying down, standing, grazing, walking or running. They achieve a cross-validated classification accuracy of 86%. This study is different from ours in terms of the data they use (100 Hz accelerometry), the amount of labelled data (over 4000) and the type of classification they perform.

Schreven et al. [10] studied nesting behaviour in Arctic-breeding geese. They propose a method for inferring nesting cycle stages (migration, pre-nesting, nesting and post-nesting) and the number of nesting days using a combination of accelerometer and GPS data. They achieve a near-perfect accuracy of 99%. Thiebault et al. [11] studied Cape gannets using acoustic data in a supervised setting to predict whether a bird is flying, floating on water or diving. They also achieve a near-perfect accuracy of 98%.

One study [12] also proposes a procedure for deriving trajectory embeddings from a jaguar movement data set [13]. This study uses long-term GPS data (with an average span of 9.8 months) with various intervals of between 0.5 to 24 hours, but they do not perform any form of classification. Rather their goal is to analyse unstructured data. This is different from our context in the sense that we only consider a few minutes of high-activity GPS data.

III. THE DATA

The data used in this work is generated by a collar fixed on an animal's neck. These collars continuously measure acceleration and, when certain parameters are met, an alarm is triggered which transmits GPS data via GSM mobile communication.

A. GPS TRAJECTORIES AND CLASSES

In our setting, a GPS trajectory refers to a time series of latitude and longitude values with a 30-second interval between points. These trajectories are recorded directly after an alarm is triggered by the device. Alarms are triggered based on onboard accelerometer data, which is not recorded. An alarm can be caused by a myriad of reasons, ranging from theft to Scrapie disease [14]. For this work, we define four main classes:

1) **Theft**: humans trying to steal livestock.
2) **Predation**: predator animals hunting livestock. These are mainly jackals but also include wild dogs, lynxes and leopards.
3) **Own-handling**: workers on the farm handling the livestock in day-to-day operations.

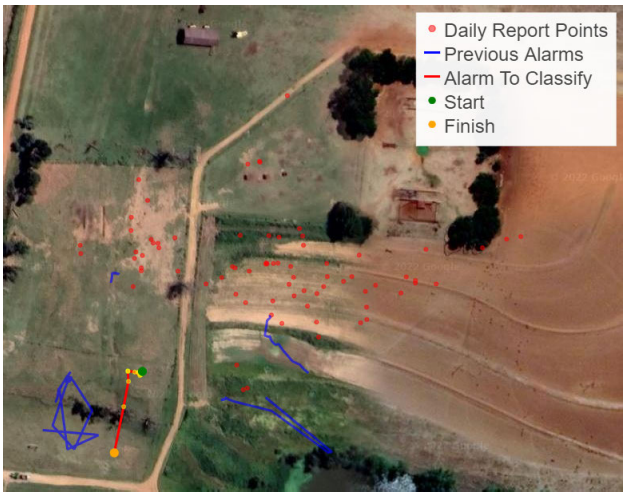


FIGURE 3. An example of all available data for a single unit, shown on a Google Maps satellite view. The red dots are GPS points reported each day and are not part of a trajectory. The blue trajectories are alarms that were previously recorded. The red trajectory is the trajectory that is currently being considered for classification.

4) **Other**: miscellaneous reasons which do not fall in the categories above. These alarms are uncommon and non-emergency phenomena such as an animal frightened by lightning strikes.

Figure 3 shows an example of all available data for a single unit. In addition to GPS trajectories resulting from alarms, we have access to a daily GPS data point that the units transmit, illustrated as red dots in Figure 3. This data point is transmitted at some predetermined time each day, irrespective of whether an alarm has been triggered.

B. AVAILABLE DATA SETS

Currently, it is difficult to acquire ground truth labels for events. The farmer must be contacted relatively soon after an alarm occurred and asked what happened. Not only is this a tedious and human-intensive task, but the acquired labels are not necessarily ground truth. A farmer might report a non-emergency when an alarm occurred, but in fact, thieves or predators could have been on the scene unknowingly, causing the alarm. Nevertheless, acquiring a small labelled data set with which the models can be evaluated is still possible. Therefore, we have three available data sets: a large unlabelled set, a small labelled validation set and a small hand-labelled training set.

TABLE 1. The class distributions of the labelled data sets.

	Theft	Predation	Own Handling	Other	Total
Validation	35	62	63	16	176
Hand-labelled	36	70	211	74	391

1) UNLABELLED DATA

A total of approximately 800 000 trajectories without labels are available. These trajectories have been captured from 2016 to the end of 2021. Roughly 500 new alarm trajectories are captured every day.

2) VALIDATION DATA

The validation data set is composed of 176 trajectories with its class distribution shown in Table 1. These labels were acquired by calling farmers within one day of the event. This data set is used for cross-validating the classifier models in the sections to come. We refer to this set as "validation data" since it is used in this work as a means to decide on a final model (other studies might have simply referred to this as the test set).

3) HAND-LABELLED TRAINING DATA

This data set was developed by a human expert hand-labelling trajectories by looking at the data as seen in Figure 3, without calling the farmers. We consider this data set to be too noisy to train on directly in a conventional purely supervised learning approach. However, we show that when using this data in a semi-supervised approach, by jointly training an autoencoder and classifier, performance is improved. One major advantage of developing a hand-labelled data set is the speed at which labels can be acquired. The class distribution of this data set is shown in Table 1.

C. HUMAN PERFORMANCE

In order to get an idea of to what extent classification is possible, we evaluate the performance of the same human expert that labelled the data set in Section III-B3. The expert is asked to classify the whole validation data set described in Section III-B2. The expert is allowed to view all available data, including a Google Maps satellite view, as shown in Figure 3. The resulting confusion matrix between the true labels and the expert's predicted labels is shown in Figure 4. We see that classification is in fact possible to some degree, with a total accuracy score of 73%. The class-specific scores are compared to the supervised and semi-supervised approaches in Sections IV and VI, respectively. From the confusion matrix, it is also clear that it is difficult to distinguish between predation and theft.

D. OBSTACLES

As in any real-world setting, the data can be highly irregular and unpredictable. In this context, the following factors have a major influence on the quality of the data and cannot be avoided in any obvious way.

1) GPS SENSOR

The GPS sensor has an accuracy of approximately 5 to 20 meters and is heavily influenced by signal strength and the position of the unit on the animal. Poor signal strength can result in unpredictable jumps in a trajectory. All other GPS

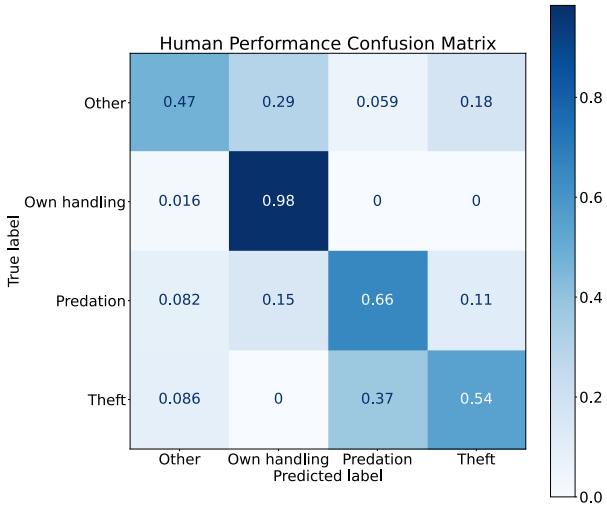


FIGURE 4. A horizontally normalised confusion matrix of human performance on the validation data, as described in Section III-B2.

obstacles apply as well, such as dilution of precision. Due to this lack of accuracy and the onboard acceleration data, it is quite possible that it may seem as if an animal is moving around when in fact, it is standing still.

2) GSM SIGNAL

The device uses GSM mobile communication to transmit data. Some data points are lost when GSM signal strength is insufficient, resulting in time jumps in the trajectory. A reduction of data points in an already sparse trajectory results in a reduction of important information. Farms can have excellent signal in one area, but poor signal in another area. Remember that these devices are typically used in remote areas.

3) TIME INTERVAL

Time irregularity is almost certain for each trajectory. As previously mentioned, time jumps (often up to a few minutes) occur if GSM signal strength is poor. In addition, by design, a new data point is transmitted immediately when the conditions for a new trigger are met. This results in a time series with compact and sparse parts in the same sequence, making it even more difficult to compare similar trajectories.

E. PROCESSING RAW DATA

The GPS values are processed to produce a distance, time, speed and angle channel for each trajectory. By design, acceleration is not included since the low sampling frequency won't allow for accurate values. More formally, we have a sequence of GPS points $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_T$, with each $\mathbf{p}_i = [\text{lat}, \text{lng}, t]$ where lat is longitude, lng is longitude and t is the timestamp. Such a sequence is illustrated in Figure 3 as the red line with its yellow dots representing the GPS points. From this sequence we produce a new feature time series $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_T$. As illustrated in Figure 5, each of these features within $\mathbf{z}_i$ is determined according to the following equations:

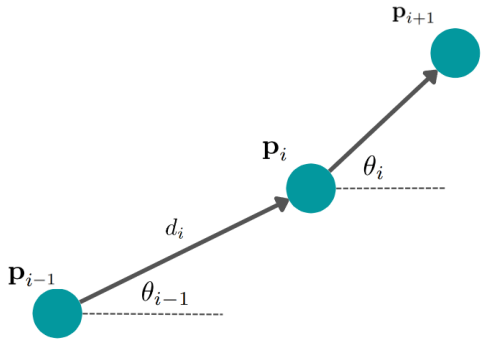


FIGURE 5. An example of GPS points in a trajectory.

$$\mathbf{z}_i = \begin{bmatrix} d_i \\ \Delta t_i \\ s_i \\ \Delta\theta_i \end{bmatrix} \tag{1}$$

$$d_i = \text{GeoDist}(\mathbf{p}_i[\text{lat}, \text{lng}], \mathbf{p}_{i-1}[\text{lat}, \text{lng}]) \tag{2}$$

$$\Delta t_i = p_i[t] - p_{i-1}[t] \tag{3}$$

$$s_i = \frac{d_i}{\Delta t_i} \tag{4}$$

$$\Delta\theta_i = \theta_i - \theta_{i-1} \tag{5}$$

Here d_i denotes the geographical distance between two GPS points, Δt_i is the difference in consecutive timestamps, s_i is the speed between two GPS points and $\Delta\theta_i$ is the difference in angle between two consecutive points. The result is a vector time series with T data points. Each trajectory $\mathbf{x}^{(n)}$ is then denoted as

$$\mathbf{x}^{(n)} = \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \ldots & \mathbf{z}_T \end{bmatrix} \tag{6}$$

with the superscript (n) indicating the n^{th} item in a data set. In Section III-F (directly below) we use this vector time series to derive a single fixed-dimensional feature vector representation for an entire sequence. In Section V we use this vector time series as input to an autoencoder which then learns to produce a latent fixed-dimensional representation of a sequence.

F. FEATURE ENGINEERING

One approach that we can follow in order to apply machine learning techniques to the data, is to manually produce a fixed-dimensional representation of each variable-length trajectory. The following is an exhaustive list of features that we derive from the available data:

1) **Total distance**: The total distance travelled during an alarm.
2) **Peak speed**: The peak speed between two consecutive points in a trajectory.

3) **Peak angle change**: The peak change in angle $\Delta\theta$ between two consecutive points in a trajectory.
4) **Duration**: The total duration of an alarm in minutes.
5) **Average speed**: The average speed between data points in a trajectory.
6) **Average angle change**: The average change in angle $\Delta\theta$ between data points in a trajectory.
7) **Displacement**: The distance between the start and the end of a trajectory.
8) **Velocity**: The velocity of a trajectory is the displacement divided by the duration.
9) **Straightness**: Straightness can be calculated as the total displacement divided by the total distance travelled. A value of 1 is a perfectly straight trajectory.
10) **Time of day**: The time of day at which the alarm occurred. This is a float value, meaning that a time of 6:30 would be represented as 6.5.
11) **Cosine time of day**: Applying the cosine function to the time of day allows for continuous indication of the time of day. Suppose one alarm occurs at 23:59 and another alarm at 00:01. Numerically these values are very far from one another, when in fact they are only 2 minutes apart. Formally this is expressed as $t_{\cos} = \cos(2\pi \frac{t}{24})$, resulting in values between -1 (midday) and 1 (midnight).
12) **Daily-report nearest neighbour distance**: The average distance of the nearest daily-report data point to each data point in the trajectory. This is a good indicator of whether a unit is in familiar territory or not.
13) **Alarm nearest neighbour distance**: The average distance of the nearest historic alarm data point to each data point in the trajectory. A value of zero would indicate that the exact trajectory has been travelled before. This is a good indicator of recurring own-handling alarms.
14) **Nearest time of day cluster**: Figure 6 shows the time of day distribution of all recorded alarms. We see two bumps around 6:30 am and 16:00 pm, a typical time when farmers would work with their animals. However, farmers' schedules differ, some farms show bumps centred around 8:00 am and 17:00 pm, for example. K-means (with $K = 2$) is performed on a unit's historic alarm times to learn recurring alarm times. The value for this feature is equal to the distance between the current alarm's time of day and the closest cluster centre. This is a good indicator of recurring own-handling alarms.
15) **Percentage of close alarms**: Percentage of historic alarms within 2 hours of the current alarm.
16) **Average number of alarms per day**: The average number of alarms that are triggered each day for a unit.
17) **Time since last alarm**: The time in minutes since the previous alarm was triggered.
18) **Animal type 1**: This value is 1 if the type of animal on which the unit is mounted is a sheep and zero otherwise.
19) **Animal type 2**: This value is 1 if the type of animal on which the unit is mounted is cattle and zero otherwise.
20) **Trigger count**: The number of trigger events in a trajectory. Although an alarm has started, the unit's algorithm continues to report new triggers within the current alarm which extends the tracking time.

Taken together, each trajectory $\mathbf{x}^{(n)}$ can now be represented with a 20-dimensional vector.

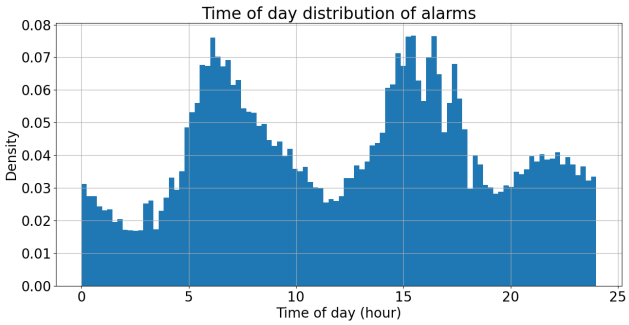


FIGURE 6. The histogram of which hour of the day alarms occurred.

IV. SUPERVISED LEARNING

In this section, we evaluate a supervised approach for classifying events. We compare various supervised models and report their cross-validation performance on the validation set described in Section III-B2. The models are trained on the fixed-dimensional representations as discussed in Section III-F.

A. MODELS

We evaluate classifier models from the Scikit-learn API [15] as well as CatBoost [16] and XGBoost [17]. Since we have little labelled data, we try to constrain the models in such a way as to prevent overfitting. For the tree-based models, we restrict the maximum depth of the tree and for the other models, we add a large regularisation penalty factor. The models with their hyper-parameters are listed below:

- **K-nearest neighbours**: number of neighbours, $K = 5$.
- **Logistic regression**: L2 regularization strength, $\alpha = 1$.
- **C-Support Vector Machine**: L2 regularization strength, $\alpha = 1$.
- **Multi-layer Perceptor (MLP)**: L2 regularization strength, $\alpha = 1$.
- **Decision tree**: Maximum tree depth = 3.
- **Random forest**: Maximum tree depth = 5.
- **Gradient boosting**: Maximum tree depth = 5.
- **AdaBoost**: Maximum tree depth = 5.
- **CatBoost**: Maximum tree depth = 5.
- **XGBoost**: Maximum tree depth = 5.

These hyper-parameters were tuned to produce the best mean cross-validation performance. The models described here are also used in our semi-supervised approach in Section VI.

TABLE 2. A comparison of class-specific F_1 scores (expressed as a percentage) for the top performing models for the supervised approach.

Class	Theft	Predation	Own-Handling	Other
Logistic Regression	**49.2**	**60.8**	89.0	5.8
MLP	47.2	59.9	**89.3**	9.8
AdaBoost	41.5	54.6	86.2	**13.0**
Human Expert	59.4	69.6	89.2	47.1
Random Baseline	20.0	24.7	26.7	10.0

B. RESULTS

To evaluate the models, we perform 15-fold cross-validation on the validation data set described in Section III-B2, with the features described in Section III-F. The top-performing models are logistic regression, MLP, and AdaBoost with accuracies of 68%, 66%, and 55%, respectively. However, in the remainder of this paper, we focus on metrics other than accuracy, since this can be misleading given the major class imbalances (see Section III-B). We rather opt for class-specific F_1 scores, defined as the harmonic mean of the recall and precision for a certain class. Class-specific recall and precision are defined as:

$$R_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FN}_c} \tag{7}$$

$$P_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FP}_c} \tag{8}$$

where the subscript c denotes a certain class. Class-specific metrics also allow us to evaluate classification performance on a class-by-class basis. This is important because we are more interested in identifying some classes than others. Concretely, classifying theft events is the top priority. Table 2 shows the resulting class-specific F_1 scores for the top-performing models. As can be seen by looking at Table 2, these models are successful in distinguishing own-handling alarms. However, the models are not as good as a human expert at classifying the remaining three classes. They are especially poor at classifying the other class. To a certain degree, this could be expected since the other class is somewhat noisy and we do not expect this class to have an inherent structure to it.

V. UNSUPERVISED TRAJECTORY EMBEDDINGS

We have a large corpus of unlabelled trajectories, as described in Section III-B1. In the preceding section, we ignored this data when training supervised models. To leverage this data, we need to incorporate unsupervised learning. In our previous work [4], we showed that classes can be distinguished to some degree in a purely unsupervised clustering approach, which incorporates fixed-dimensional autoencoder embeddings. In Section VI these unsupervised trajectory embeddings are combined with the list of features described in Section III-F to form a semi-supervised classifier model.

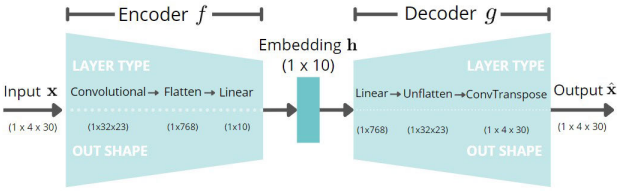


FIGURE 7. The architecture of the convolutional-deconvolutional autoencoder. The model takes a four-channel, one-dimensional input as described in equation 6, encodes it to a ten-dimensional vector and then decodes it to reproduce the input. Layer types and output shapes are shown. The convolutional component has three 1-D convolution layers with 8, 16 and 32 filters respectively, each followed by a ReLU layer. All filters have a size of 3 and a stride of 1.

A. MODEL: TRAJECTORY EMBEDDINGS

In order to produce fixed-dimensional trajectory embeddings, we implement a convolutional-deconvolutional autoencoder (AE) with the goal of extracting valuable features from the GPS data which can be used for downstream classification. The input to the model is the feature time series described in equation 6. By training this model to reconstruct its input through a lower-dimensional compressed representation, the hope is that the latent embedding would capture distinguishing features. We encourage the reader to refer to our previous work [4] where this approach is covered in detail.

The model's architecture is shown in Figure 7. First, the input is encoded to a fixed-dimensional space smaller than the dimensionality of the input, called the latent trajectory embedding $\mathbf{h}$, and then decoded to the original form of the trajectory. We constrict the latent embedding to a fixed 10 dimensions and limit the length of each time series $\mathbf{x}^{(n)}$ described in equation (6) to $T = 30$, which is the default length of alarms. A value of 10 for the size of the latent embedding was arrived upon based on downstream clustering metrics in development experiments. The model is trained by minimizing the mean squared error (MSE) loss between the input $\mathbf{x}^{(n)}$ and the reconstructed output $\hat{\mathbf{x}}^{(n)}$:

$$L_{\mathrm{AE}} = \frac{1}{N}\sum_{n=1}^{N} l_{\mathrm{MSE}}(\mathbf{x}^{(n)}, \hat{\mathbf{x}}^{(n)}) \tag{9}$$

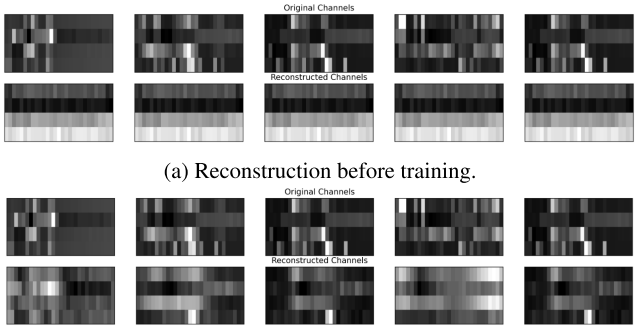
with

$$l_{\mathrm{MSE}}(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{T}\sum_{i=1}^{T} ||\mathbf{z}_i - \hat{\mathbf{z}}_i||^2 \tag{10}$$

L_{AE} is therefore the mean loss over all the training trajectories and N is the batch size. The model is trained with the Adam [18] optimiser, a batch size of 256 and a learning rate of 0.05.

B. INTERMEDIATE EVALUATION

After training the model for 100 epochs on the unlabelled data described in Section III-B1, the mean square error loss of the autoencoder on the validation and training set is 0.7 and 0.3, respectively. Figure 8 shows the reconstruction before and after training the model. The model is able to learn useful



(a) Reconstruction before training.

(b) Reconstruction after training.

FIGURE 8. Grey-scale images to show the reconstruction that the autoencoder produces for five samples (a) before and (b) after training. These samples are not seen during training. Each row in each image is a channel of the sample as described in Section III-E.

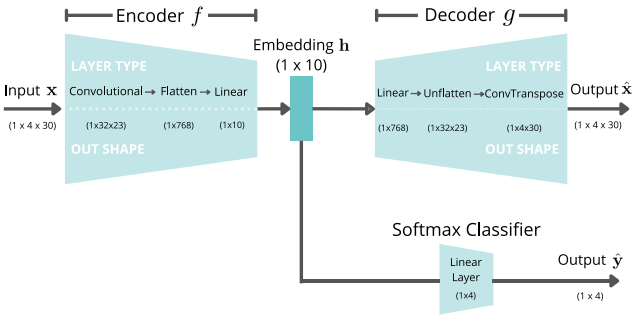


FIGURE 9. The semi-supervised model architecture. The autoencoder branch is the same as in Figure 7. The classifier branch consists of a fully connected linear layer with 4 output values, one for each class. As a result of joint training, the classifier's classification performance is much worse than the supervised and semi-supervised approaches.

features that can be used to reconstruct the input. When we perform K-means clustering (with K = 7) on the resulting embeddings, we achieve a total clustering purity of 60%, calculated on the validation set described in Section III-B2. The hope is that these extracted features, which can be used to reconstruct a trajectory, will help to classify events.

VI. SEMI-SUPERVISED LEARNING

In this section, we evaluate a semi-supervised approach for classifying events. The approach incorporates the unlabelled data in Section III-B1 as well as the hand-labelled training data in Section III-B3. Concretely, our semi-supervised approach consists of three parts:

1) Extending the unsupervised model in Section V with a classifier that is jointly trained with the autoencoder.
2) Performing K-means clustering on the resulting trajectory embeddings and assigning cluster labels. Note that we do not use the embeddings directly.
3) Training new classifier models which incorporate both the cluster labels and the list of features described in Section III-F. These models are trained and evaluated in a 15-fold cross-validation setup on the ground truth data described in Section III-B2.

A. JOINT TRAINING

We train an autoencoder and a single-layer softmax classifier jointly, with the architecture shown in Figure 9. The autoencoder is trained on the unlabelled data set as before in Section V. We use the hand-labelled training data set, as described in Section III-B3, to train the classifier. Although this data set is considered to be unsuitable for training a final classifier, the idea here is to improve the embeddings. By adding the classifier, we inject crucial information into the embeddings, which is not present in the trajectory data alone and is therefore absent in the purely unsupervised approach from Section V. Concretely, we are forcing the embeddings to capture information required for classification as well, not only for reconstruction. This model is trained with a loss function defined as

$$L_{\text{total}} = L_{\text{CLF}} + L_{\text{AE}} \tag{11}$$

with cross-entropy (CE) as the classification loss between the class label y and the predicted class distribution $\hat{\mathbf{y}}$:

$$L_{\text{CLF}} = \frac{1}{N}\sum_{n=1}^{N} l_{\text{CE}}(y^{(n)}, \hat{\mathbf{y}}^{(n)}) \tag{12}$$

with

$$l_{\text{CE}}(y, \hat{\mathbf{y}}) = -\log \frac{\exp(\hat{y}_{c=y})}{\sum_{c=1}^{C}\exp(\hat{y}_c)} \tag{13}$$

where $C = 4$ is the number of classes and N is the batch size. The reconstruction loss L_{AE} is already defined in equation (9). In equation 11, we can assign weights to the losses in order to prioritise one task above the other. These weights can then also be scheduled throughout the training process. We experimented with these techniques but saw no improvement in downstream classification performance.

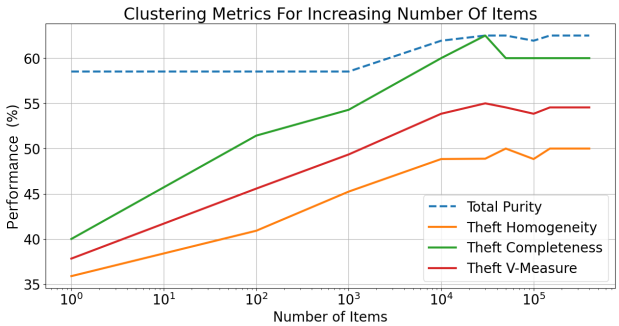


FIGURE 10. Clustering metrics after K-means is performed, for an increasing number of unlabelled training data.

B. CLUSTERING

After training the joint model, K-means clustering is performed on the resulting embeddings of the large unlabelled data set. A value of $K = 7$ was arrived upon

based on the elbow method, silhouette method [19] and the Davies-Bouldin Index [20] giving roughly the same number of clusters. These cluster labels are then used as a feature for the downstream classification models. Note that we do not use the embeddings directly. When using the embeddings directly, we do not see an increase in classification performance in comparison to the supervised approach in Section IV. However, when using the cluster labels, we are injecting the structural information of the whole unlabelled data set. This effect is visualised in Figure 10, where we see an improvement in clustering performance metrics as the number of items used for clustering increases. If we were to use the embeddings directly, we lose out on this structural information.

In terms of clustering performance metrics, we evaluate total purity and homogeneity, completeness and V-measure for the theft class. In short, homogeneity, completeness and V-measure are comparable to precision, recall and F_1 score, respectively. The semi-supervised model outperforms the purely unsupervised approach in Section V. Total purity is increased from 60% to 63% and theft V-measure increased from 47% to 55%. These results are not specifically important, they are simply an intermediate check to show that the clustering is reasonable to be used downstream.

C. CLASSIFICATION

The cluster labels, in the form of a one-hot vector, are concatenated to the feature vector described in Section III-F. We train the same models as described in Section IV-A. The classifier branch of the model performs much worse in comparison to the other classifiers. This is to be expected because we are not optimising the model for classification alone. Note that, because of the larger feature vector and the small amount of data the classifiers are trained on, we might expect the models to be more at risk of overfitting.

D. RESULTS

To evaluate the classification performance, we perform 15-fold cross-validation on the validation data set described in Section III-B2. The top-performing models achieve accuracies of 67%, 66%, and 59%, respectively. Again, we rather focus on F_1 scores, since accuracy can be misleading given the class imbalance. Table 3 shows the resulting class-specific F_1 scores for the best-performing models. The MLP model shows a significant increase in theft F_1 score and is overall the best-performing model. The model shows comparable performance to a human expert in identifying theft and own-handling alarms. However, the model is worse at identifying predation alarms and is poor at identifying other alarms.

VII. COMPARING SUPERVISED AND SEMI-SUPERVISED LEARNING

We have evaluated two approaches for classification: a supervised approach described in Section IV and a semi-supervised approach described in Section VI. In this section, we compare these two methodologies. A comparison of class-specific F_1 scores between these two approaches is shown in Table 4. Both approaches show comparable performance to a human expert in classifying own-handling events. The semi-supervised approach shows an increase in theft F_1 score in comparison to the supervised approach. Both approaches are poor at classifying other events. As previously mentioned, this is to be expected since there is no inherent structure to events from this class.

It is important to remember that, during 15-fold cross-validation, classifier models are only trained on 164 to 165 items. Due to this, we are unable to train complex classifier models because the models are heavily restricted to avoid overfitting. Therefore, we can expect even better performance once more labelled data has been acquired.

TABLE 3. A comparison of class-specific F_1 scores (expressed as a percentage) for the top performing models for the semi-supervised approach.

Class	Theft	Predation	Own-Handling	Other
Logistic Regression	48.9	57.7	88.3	10.8
MLP	**55.6**	**60.1**	**89.5**	10.5
AdaBoost	47.7	51.5	86.2	**14.8**
Human Expert	59.4	69.6	89.2	47.1
Random Baseline	20.0	24.7	26.7	10.0

TABLE 4. The class-specific F_1 scores (expressed as a percentage) for the top performing classifier for the supervised and semi-supervised approach. Human performance is also shown.

Class	Theft	Predation	Own-Handling	Other
Supervised	49.2	**60.8**	89.0	5.8
Semi-Supervised	**55.6**	60.1	**89.5**	**10.5**
Human Expert	59.4	69.6	89.2	47.1

VIII. CONCLUSION

In this work, we introduce a novel application of machine learning: using GPS trajectory data to classify livestock threats. We compare a supervised and a semi-supervised approach for classification. We show that a semi-supervised approach outperforms a supervised approach, especially in distinguishing theft.

A. GOAL: CLASSIFICATION

Our goal is the classification of livestock threats with a special emphasis on the theft class. The final semi-supervised model shows comparable performance to a human expert in theft and own-handling classification. We show that, within this low-resource setting, the classification of livestock threats is indeed possible to an extent. By deploying this model,

farmers can be provided with critical information in emergency situations.

B. FUTURE WORK

We propose to increase the GPS sampling rate so that more nuanced information can be captured in a trajectory which should, in turn, improve classification performance. More labelled data should be acquired so that more complex classifier models can be developed. Other channels of data should be investigated. Recording acceleration data could prove to be useful when combined with GPS data.

REFERENCES

[1] *Farm Attacks and Farm Murders in South Africa*. Accessed: Mar. 20, 2023. [Online]. Available: https://afriforum.co.za/wp-content/uploads/2021/09/Farm-attacks-and-farm-murders-in-South-Africa-Analysis-of-recorded-incidents-2019.pdf

[2] *The Impact of Stock Theft*. Accessed: Mar. 20, 2023. [Online]. Available: https://www.harvestsa.co.za/2021/11/12/the-impact-of-stock-theft/

[3] H. van Niekerk, "The cost of predation on small livestock in South Africa by medium-sized predators," M.S. thesis, Dept. Agricult. Econ., Fac. Natural Agricult. Sci., Univ. Free State, Bloemfontein, South Africa, 2010.

[4] U. De Swardt and H. Kamper, "How machine learning can aid South African farmers' security: Unsupervised livestock trajectory embeddings," in *Proc. South Afr. Conf. AI Res.*, 2022, pp. 54–66.

[5] S. Dabiri, C.-T. Lu, K. Heaslip, and C. K. Reddy, "Semi-supervised deep learning approach for transportation mode identification using GPS trajectory data," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 5, pp. 1010–1023, May 2020.

[6] Y. Zheng, H. Fu, X. Xie, W.-Y. Ma, and Q. Li. (2011). *Geolife GPS Trajectory Dataset—User Guide*. [Online]. Available: https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/

[7] H. Kamper, A. Jansen, S. King, and S. Goldwater, "Unsupervised lexical clustering of speech segments using fixed-dimensional acoustic embeddings," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Dec. 2014, pp. 100–105.

[8] H. Kamper, W. Wang, and K. Livescu, "Deep convolutional acoustic word embeddings using word-pair side information," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 4950–4954.

[9] S. P. L. Roux, J. Marias, R. Wolhuter, and T. Niesler, "Animal-borne behaviour classification for sheep (Dohne Merino) and rhinoceros (*Ceratotherium simum* and *Diceros bicornis*)," *Animal Biotelemetry*, vol. 5, no. 1, Dec. 2017, Art. no. 25.

[10] K. H. T. Schreven, C. Stolz, J. Madsen, and B. A. Nolet, "Nesting attempts and success of arctic-breeding geese can be derived with high precision from accelerometry and GPS-tracking," *Animal Biotelemetry*, vol. 9, no. 1, pp. 1–13, Dec. 2021.

[11] A. Thiebault, C. Huetz, P. Pistorius, T. Aubin, and I. Charrier, "Animal-borne acoustic data alone can provide high accuracy classification of activity budgets," *Animal Biotelemetry*, vol. 9, no. 1, pp. 1–16, Dec. 2021.

[12] A. Waal and A. Graaff, "Deriving trajectory embeddings from animal movement data," in *Proc. South Afr. Conf. AI Res.*, 2021.

[13] R. G. Morato et al., "Jaguar movement database: A GPS-based movement dataset of an apex predator in the neotropics," *Ecology*, vol. 99, no. 7, p. 1691, Jul. 2018.

[14] P. Greenwood, "Federal disease control—Scrapie," *Can. Veterinary J.*, vol. 43, no. 8, pp. 625–629, 2002.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011.

[16] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: Gradient boosting with categorical features support," 2018, *arXiv:1810.11363*.

[17] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.

[18] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–15.

[19] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, Nov. 1987.

[20] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.

URS J. DE SWARDT received the B.Eng. degree in electrical and electronics engineering from Stellenbosch University, South Africa, in 2021. He is currently pursuing the master's degree in electronic engineering with Stellenbosch University. He also started working as a Software Engineer at Etse Electronics, South Africa, in 2021. His research interest includes improving livestock security by using machine-learning techniques.

HERMAN KAMPER received the Ph.D. degree from The University of Edinburgh, in 2017. He did postdoctoral research with the Toyota Technological Institute at Chicago, working on multimodal machine-learning models combining speech and vision. He worked on unsupervised speech processing, specifically combining unsupervised Bayesian and neural models with The University of Edinburgh. He is currently an Associate Professor in electrical and electronic engineering with Stellenbosch University, South Africa. His interests include developing methods that would allow machines to acquire speech and language processing capabilities with as little supervision as possible. This would enable language technology in low-resource settings and could tell us something about how humans acquire language.