How Machine Learning Can Aid South African Farmers' Security: Unsupervised Livestock Trajectory Embeddings*

Urs de Swardt[0000−0002−2715−8348] and Herman Kamper[0000−0003−2980−3475]

E&E Engineering, Stellenbosch University, South Africa
{21948828,kamperh}@sun.ac.za

Abstract. The National Stock Theft Prevention Forum estimates annual losses of up to R3 billion owing to stock theft on South African farms. These concerns sparked innovative technologies in the security industry, one of which is a device for livestock that transmits GPS data when an animal is in distress. In this paper, time series machine learning techniques are applied to real-world livestock GPS trajectories. Our main goal is to distinguish between four categories of trajectories: theft, predation, own handling and other. We lay special emphasis on distinguishing theft-alarms from the rest since these have direct implications for the safety and financial sustainability of farmers. We have access to a large number of trajectories recorded over the last six years. Unfortunately, these trajectories are not labelled with the four categories. In this unsupervised setting, we propose a livestock trajectory embedding (LTE) model as a feature extractor for downstream clustering. The LTE model has a convolutional-deconvolutional architecture and is trained as an autoencoder to reconstruct its trajectory input. The proposed approach achieves a purity of 59.66%. We also show that the model produces a purity of 80.11% when only considering emergencies vs non-emergencies. We hope that the clusters predicted by our model could be used in downstream classification systems to provide critical information to farmers in emergency situations. Based on the results in this paper, we recommend that for future work, the upstream data resolution should be increased in order to increase overall performance.

Keywords: GPS trajectory · livestock movement · unsupervised learning · time series embeddings · IoT.

1 Introduction

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 be an annual loss of 13% for production animals [12]. 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

* Supported by FarmRanger.

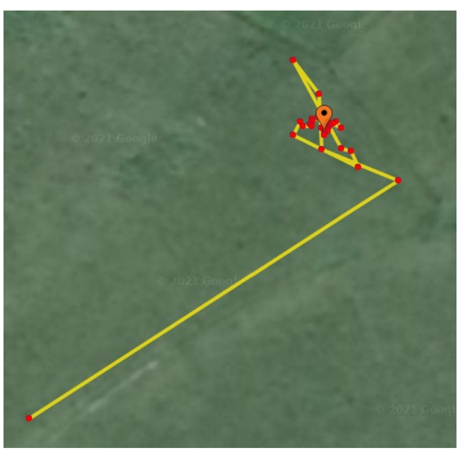
livestock from theft and predation.[1] A single unit is attached to an animal in a flock or herd. The unit monitors acceleration and when certain conditions are met,[2] it triggers an alarm that transmits GPS data to the owner. FarmRanger has recorded this GPS data since 2016, with just short of a million alarms to date. However, it is unknown what really happened during each of these events. It could have been theft, predation or one of several other possibilities that can cause rapid movement.

Figure 1 shows two examples of what a user would typically see on FarmRanger's app. 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 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 only 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[3] from one another? Doing so would equip a FarmRanger client with the

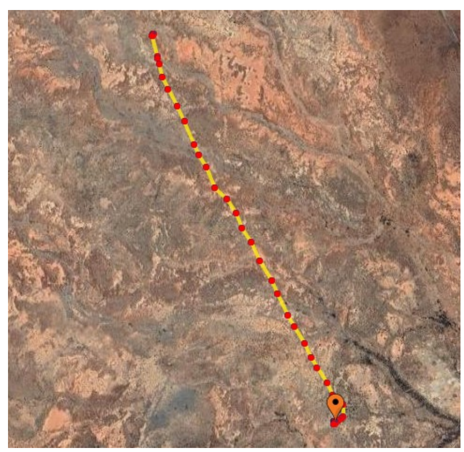
[1] More details about FarmRanger can be found at www.farmranger.co.za.

[2] 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.

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



(a) Example of a jackal attack.



(b) Example of theft.

Fig. 1: The GPS data points of two examples of ideal scenarios. One can see that in the case of (a) there is random movement without the sense of 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.

necessary knowledge to properly prepare for an emergency and ultimately help keep our farmers safe. As a secondary goal, we would like to distinguish emergency events (theft and predation) from non-emergency events (own-handling and other).

In this paper, we introduce a new application of time series machine learning techniques to address the aforementioned problem. Concretely, we propose a livestock trajectory embedding (LTE) model to act as a feature extractor which can be used for K-means clustering of trajectories. The LTE model is a convolutional-deconvolutional autoencoder which is trained to reconstruct its trajectory input. The model encodes a given trajectory to a fixed-dimensional feature space which is in turn decoded to the original trajectory. This fixed-dimensional encoding is a trajectory's embedding. Our model is compared to two other approaches for feature extraction by performing K-means clustering and calculating purity and other clustering metrics on the resulting clusters. We show that LTE outperforms the other two baseline approaches.

2 Related Work

One other machine learning problem that also utilises GPS trajectories, is the task of classifying mode-of-transport. This means 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 [3]. This is similar to the problem that we are interested in, in the sense that extracting useful features from the raw GPS points is crucial for accurate classification. However, one major difference is that this is typically framed as a semi-supervised problem with labelled and unlabelled data [17]. In addition, the time interval between data points is relatively small (1-5 seconds) for mode-of-transport classification, 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 [3]. In this model, an autoencoder and a classifier are trained jointly with weighted losses that can be tuned. The classifier is simply a softmax layer added to the encoder. We follow a similar but fully unsupervised approach for our LTE model.

Our LTE model 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 LTE model in the sense that they produce a fixed-dimensional representation of a time series — in this case 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. In the same way, the aim of the LTE model is to produce fixed-dimensional embeddings for GPS time series where similar trajectories are close to one another. As in [8], we use a convolutional neural network as the basis for our LTE model. Other acoustic word embedding models have also used recurrent neural networks [6, 16], but we leave a comparison between these two network types for future work. This work in the speech processing area precedes the work in mode-of-transport classification mentioned above.

3 Data: Livestock Trajectories

A livestock 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 an onboard accelerometer.[4] An alarm can be caused by a myriad of reasons, ranging from theft to "Mad Sheep" disease. 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.
4. **Other**: miscellaneous reasons which do not fall in the above categories. These alarms are uncommon and non-emergency phenomena like the previously mentioned "Mad Sheep" disease.

3.1 Data Sets

Currently, it is troublesome to acquire 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. Nevertheless, it is still possible to acquire a small labelled data set with which the models can be evaluated. We, therefore, have two available data sets, a large unlabelled training set and a small labelled validation set.

Training Data. A total of approximately 800 000 trajectories are available in the training set with no labels available. FarmRanger records around 500 new alarms every day.

[4] Accelerometer data is not recorded.

Table 1: The class distribution for the validation data set.

	Theft	Predation	Own handling	Other
Count	35	62	63	16

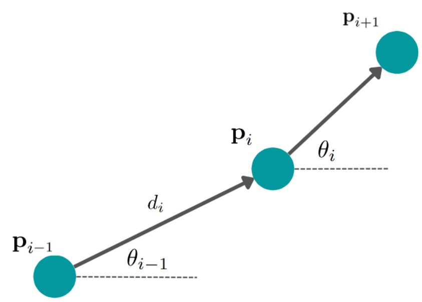


Fig. 2: An example of three GPS data points in a trajectory. Each point $\mathbf{p}_i$ has a corresponding latitude, longitude and time value.

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. The true, real-world distribution of classes remains unknown, therefore it is impossible to know if the validation set provides a true representation of the data in terms of the class distribution.

3.2 Processing Raw Data

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 $\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_T$, with each $\mathbf{p}_i = [\text{lat}, \text{lng}, t]$. From this sequence we produce a new feature time series $\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_T$. Each of these features within $\mathbf{z}_i$ is determined as in Figure 2, according to the following equations:

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

$$d_i = \text{GeoDist}(p_i[\text{lat}, \text{lng}], p_{i-1}[\text{lat}, \text{lng}]) \tag{2}$$

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

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

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

where GeoDist denotes the geographical distance between two GPS points. The result is a four-channel one-dimensional vector time series. We limit the length of the time series to $T = 30$ since this is the default recorded length for alarms. If a trajectory has less than 30 data points, it is padded with zeros. Each trajectory $\mathbf{x}^{(n)}$ is then denoted as

$$\mathbf{x}^{(n)} = \left[\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_T\right], T = 30$$

with the superscript (n) indicating the n^{th} training or validation trajectory. The whole feature vector is scaled to have zero mean and unit variance.

3.3 Obstacles

As in any real-world setting, the data can be highly irregular and unpredictable. In this context, the following factors influence the quality of the data in a major way.

GPS Sensor. The GPS sensor has a best accuracy of approximately 5 meters and is heavily influenced by signal strength. Poor signal strength can result in unpredictable jumps in a trajectory. All other GPS obstacles apply as well, such as dilution of precision (DOP).

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. Farms can have excellent signal in one area, but poor signal in another area.

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.

4 Model: Livestock Trajectory Embeddings

Our livestock trajectory embedding (LTE) model is heavily inspired by [7, 8, 3]. Concretely, it is a convolutional-deconvolutional autoencoder with the architecture shown in Figure 3. In essence, an autoencoder is an unsupervised technique which tries to reconstruct its input with the aim of capturing valuable information in the process. First, the input is encoded to a fixed-dimensional space smaller than the dimensionality of the input,[5] called the latent embedding $\mathbf{h}$, and then decoded to the original form of the trajectory. We constrict the latent embedding to a fixed 10 dimensions.[6] By training this model to reconstruct its input through a lower-dimensional compressed representation, the hope is that the latent embedding would capture meaningful features that can be used in downstream tasks.

Formally, the reconstruction $\hat{\mathbf{x}}$ can be described by:

$$\hat{\mathbf{x}} = g(f(\mathbf{x})) \tag{6}$$

[5] Technically this is called an under complete autoencoder [5].

[6] The size of the latent embedding was fine-tuned to 10 based on the evaluation metrics in Section 5.2, calculated on the validation set.

where f is the encoder architecture producing $\mathbf{h}$ from the input $\mathbf{x}$ and g is the decoder architecture producing $\hat{\mathbf{x}}$ from $\mathbf{h}$. The model is trained by minimizing the mean squared error (MSE) loss function:

$$L = \frac{1}{N}\sum_{n=1}^{N} l(\mathbf{x}^{(n)}, \hat{\mathbf{x}}^{(n)}) \tag{7}$$

with

$$l(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{T}\sum_{i=1}^{T} ||\mathbf{z}_i - \hat{\mathbf{z}}_i||^2 \tag{8}$$

L is therefore the total loss, $\mathbf{x}$ is the input and $\hat{\mathbf{x}}$ is the reconstruction of the input. In our case, the LTE model is trained with the Adam [9] optimizer, a batch size of 256 and a learning rate of 0.05 for 300 epochs on the training data described in Section 3.1. After training the LTE model and embedding all the trajectories, we apply K-means clustering to cluster the trajectories. A value of $K = 7$ was arrived upon based on the Elbow Method, Silhouette Method [14] and the Davies-Bouldin Index [4] giving roughly the same number of clusters.

5 Experimental Setup

The goal of extracting fixed-dimensional features from the raw GPS data is to cluster similar trajectories. We consider two baseline approaches, both of which also produce fixed-dimensional representations of a trajectory. To compare the quality of these representations to the proposed LTE method, we perform K-means clustering on the respective representations and then calculate purity and other clustering metrics on the validation data.

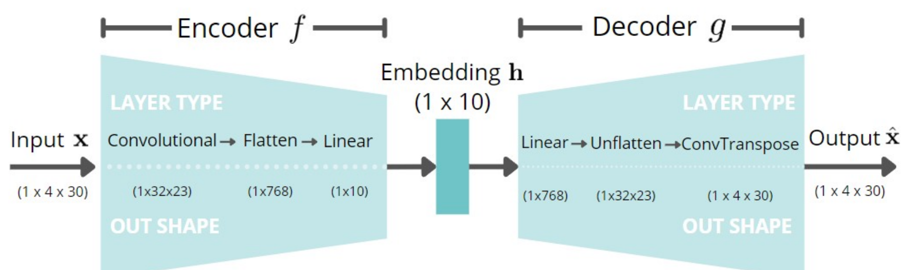


Fig. 3: The architecture of the convolutional-deconvolutional autoencoder. The model takes a four-channel, one-dimensional input, encodes it to a ten-dimensional vector and then decodes it to reproduce the input. Layer types and output shapes are shown. The convolutional component has three 1-D convolution 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.

5.1 Baseline Approaches

Two baseline approaches are implemented in order to produce features for clustering. The resulting feature vectors are scaled to have zero mean and unit variance and are then clustered by performing K-means with $K = 7$. We also report random assignment as a third baseline.

Dynamic Time Warping. Dynamic Time Warping (DTW) is a common algorithm used to calculate a distance metric (or alignment cost) between two time series with variable lengths [15]. We follow the method in [10] to produce fixed-dimensional features from trajectories. 100 trajectories are chosen at random from the training data set as exemplars to serve as a *reference set*. The DTW distance between each trajectory and each exemplar in the reference set is then calculated to produce 100 features for each trajectory. These fixed-dimensional representations can now be clustered and metrics can be calculated from the validation set.

Feature Engineering. Feature engineering is the process of a human designing features based on an understanding of the context of the task and the data. For this purpose, we engineer five intuitive features to summarize the whole trajectory:

1. The peak speed.
2. The average speed.
3. The average angle change between points.
4. The *straightness*, calculated as total displacement divided by total distance travelled. A value of 1 is a perfectly straight trajectory.
5. The time of the day when the alarm occurred. The cosine function is used to convert the hour of the day to a value between -1 and 1 where -1 is the middle of the day and 1 is the middle of the night.

The result is a 5-dimensional vector for each trajectory.

5.2 Evaluation

The LTE model will be evaluated in three ways namely inspection, cluster purity and theft V-measure.

Inspection. We inspect the LTE model by using various techniques to visualise:

- The reconstruction of the autoencoder.
- The embedded space.
- Clustering.

Cluster Purity. Given N observations, K clusters and C classes, total cluster purity is defined as:

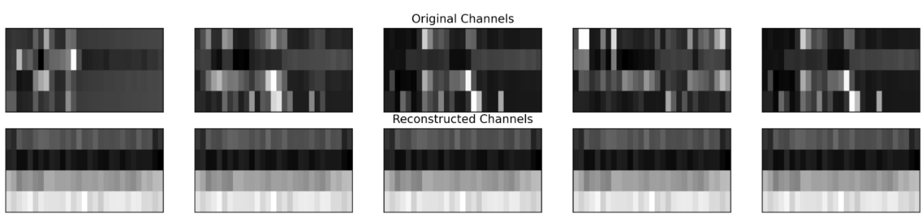
$$\frac{1}{N}\sum_{k=1}^{K}\max_{c\in C}\{c\cap k\} \tag{9}$$

Two different purity scores are considered. First, total purity for all classes, as described in (9). Second, the total purity when only evaluating emergencies (theft and predation) versus non-emergencies (own-handling and other) since this is also a valuable distinction.

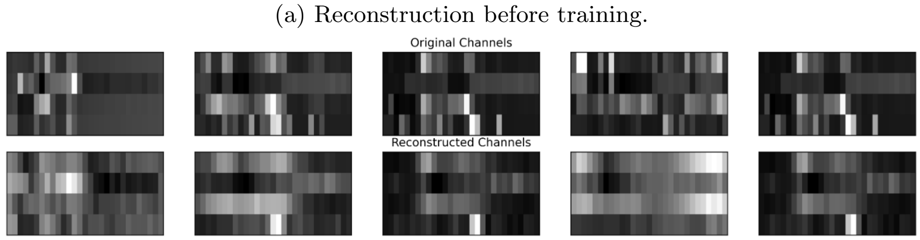
Homogeneity, Completeness and V-measure for Theft. We are especially interested in distinguishing theft from other alarms. Therefore, the V-measure for theft will be evaluated. V-measure is the harmonic mean of homogeneity and completeness [13]. Homogeneity gives an indication of how homogeneous (or pure) clusters are whereas completeness gives an indication of the tendency of a class to belong to the same cluster. Homogeneity, completeness and V-measure are similar to precision, recall and F-score, respectively. These metrics are derived for a single class (theft) as follows:

$$\text{homogeneity} = \max_{k\in K}\left\{\frac{c_{\text{theft}}\cap k}{n_{\text{theft},k}}\right\} \tag{10}$$

$$\text{completeness} = \max_{k\in K}\left\{\frac{c_{\text{theft}}\cap k}{n_{\text{theft}}}\right\} \tag{11}$$



(a) Reconstruction before training.



(b) Reconstruction after training.

Fig. 4: 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 3.

$$\text{V-measure} = 2 \times \frac{\text{homogeneity} \times \text{completeness}}{\text{homogeneity} + \text{completeness}} \tag{12}$$

Note that this way of calculating these metrics for a single class is not the same as first proposed by [13], but it has the same goal and descriptive value. Theft V-measure specifically gives an indication of how well we can isolate theft events.

6 Results

6.1 Autoencoder Reconstruction

Although the quality of the reconstruction of the input is important, the ultimate goal is not to reproduce the input but to embed useful features. After training for 100 epochs, the mean square error loss of the autoencoder on the validation and training set is 0.7 and 0.3 respectively. Figure 4 shows the reconstruction before and after training the model. It is clear that the model is able to learn useful features that can be used to reconstruct the input.

6.2 UMAP Visualisation

A dimension reduction technique, Uniform Manifold Approximation and Projection (UMAP) [11], is used to visualise the ten-dimensional embedded space.

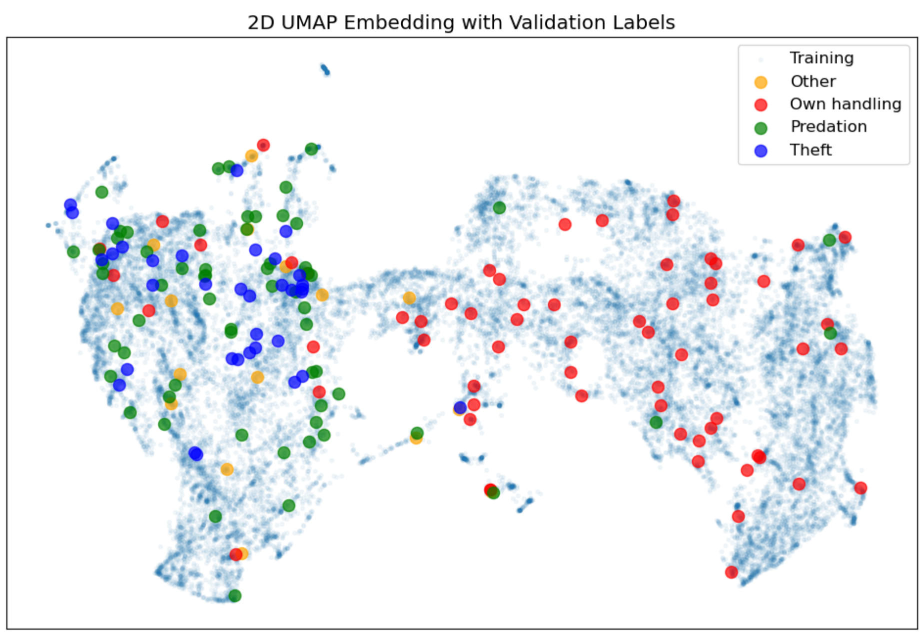


Fig. 5: A two-dimensional scatter plot of a UMAP embedding performed on the ten-dimensional encoded trajectories. The small blue dots are training data while the validation data is colour coded.

UMAP allows us to inspect the structure of the higher-dimensional space on a two-dimensional plot as seen in Figure 5. We can see two major clusters forming, one on the right with the majority of the own handling trajectories and one on the left with a medley of predation and theft trajectories.

6.3 Clustering

K-means clustering is performed on the embedding and the resulting clusters are shown in Figure 6. Three major clusters can be seen. Cluster 2 is an almost pure own handling cluster. The majority of cluster 3 is predation. Cluster 4 has equal counts for predation and theft. We also see some almost-empty clusters which consist of outliers — these are typically due to the obstacles listed in Section 3.3. Although the clustering is not perfect, it still provides valuable information and shows that distinctions can be made.

6.4 Quantitative Results

The two baseline approaches as described in Section 5.1 are implemented to produce features for clustering. After K-means (with $K = 7$) clustering is performed, the metrics as described in Section 5.2 are calculated and documented in Table 2. As a sanity check, the metrics are also calculated for random cluster assignment.

Feature engineering performs relatively poorly with similar results to random assignment. Although purity scores for DTW and LTE are comparable, there is a large distinction in theft V-measure. We can therefore conclude that LTE is the superior approach since it outperforms the other approaches in all metrics. It is also clear that a better distinction can be made between emergencies and non-emergencies. This distinction is valuable because only emergencies require a response from the farmer.

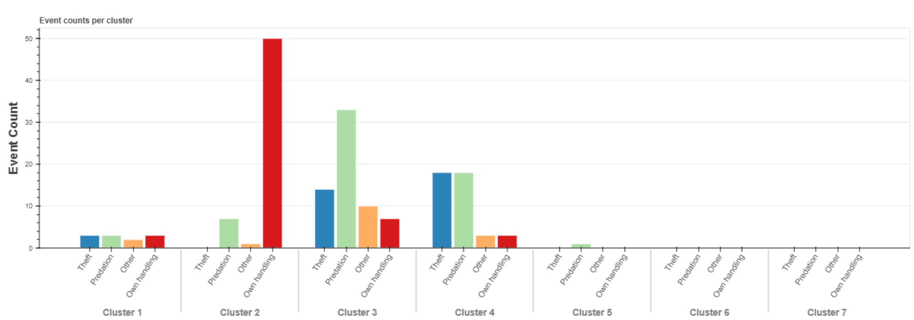


Fig. 6: K-means (with $K = 7$) clustering results. The y-axis shows the counts of classes and the x-axis shows each cluster.

Table 2: Quantitative Results (in percentage, %) for LTE, DTW, feature engineering and random assignment. For all the shown metrics, higher is better.

	Total Purity	Emergency vs Non-Emergency Purity	Theft Homogeneity	Theft Completeness	Theft V-Measure
Random Assignment	41.60	57.93	27.99	25.51	26.57
Feature Engineering	43.75	58.52	27.27	25.71	26.47
DTW	56.25	77.84	27.08	37.14	31.33
LTE	**59.66**	**80.11**	**42.86**	**51.42**	**46.75**

7 Conclusion

This paper introduces a new application of time series machine learning techniques. Concretely, we propose a convolutional-deconvolutional autoencoder to produce livestock trajectory embeddings (LTE). LTE is compared to feature engineering and a dynamic time warping (DTW) approach by performing K-means clustering on extracted features and calculating key metrics. LTE outperforms the other approaches on all metrics.

Although not perfect, we suggest that our approach is capable of providing valuable embeddings which can be used for downstream classification. Improving upstream data quality in terms of sampling frequency should reveal more information about a trajectory which should, in turn, improve embeddings. The fact that events can be distinguished in an unsupervised fashion suggests that investing in acquiring labels for events might be worthwhile, so that semi-supervised or supervised techniques can be incorporated in future work.

By utilising the model proposed in this paper, downstream classification would be able to provide critical information to farmers when they need it most.

References

1. Farm attacks and farm murders in south africa, 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, https://www.harvestsa.co.za/2021/11/12/the-impact-of-stock-theft/
3. Dabiri, S., Lu, C.T., Heaslip, K., Reddy, C.K.: Semi-supervised deep learning approach for transportation mode identification using GPS trajectory data. 2020 IEEE Transactions on Knowledge and Data Engineering (TKDE)
4. Davies, D.L., Bouldin, D.W.: A cluster separation measure. 1979 IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), http://www.deeplearningbook.org

6. Kamper, H.: Truly unsupervised acoustic word embeddings using weak top-down constraints in encoder-decoder models. In: 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)
7. Kamper, H., Jansen, A., King, S., Goldwater, S.: Unsupervised lexical clustering of speech segments using fixed-dimensional acoustic embeddings. In: 2014 IEEE Spoken Language Technology Workshop (SLT)
8. Kamper, H., Wang, W., Livescu, K.: Deep convolutional acoustic word embeddings using word-pair side information. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)
9. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: 2014 International Conference on Learning Representations (ICLR)
10. Levin, K., Henry, K., Jansen, A., Livescu, K.: Fixed-dimensional acoustic embeddings of variable-length segments in low-resource settings. In: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)
11. McInnes, L., Healy, J., Saul, N., Großberger, L.: Umap: Uniform manifold approximation and projection. 2018 Journal of Open Source Software (JOSS)
12. van Niekerk, H.: The cost of predation on small livestock in South Africa by medium-sized predators. Ph.D. thesis, University of the Free State
13. Rosenberg, A., Hirschberg, J.: V-measure: A conditional entropy-based external cluster evaluation measure. In: 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)
14. Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. 1987 Journal of Computational and Applied Mathematics (JCAM)
15. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. 1978 IEEE Transactions on Acoustics, Speech, and Signal Processing (TASSP)
16. Settle, S., Livescu, K.: Discriminative acoustic word embeddings: Tecurrent neural network-based approaches. In: 2016 IEEE Spoken Language Technology Workshop (SLT)
17. Zheng, Y., Fu, H., Xie, X., Ma, W.Y., Li, Q.: Geolife GPS trajectory dataset - User Guide, geolife GPS trajectories 1.1 edn. (July 2011), https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/