Improving Smartphone based Transport Mode Recognition using Generative Adversarial Networks

Lukas Günthermann, Andrew Philippides, and Daniel Roggen

Abstract Wearable devices such as smartphones and smartwatches are widely used and record a significant amount of data. Labelling this data for human activity recognition is a time-consuming task, therefore methods which reduce the amount of labelled data required to train accurate classifiers are important. Generative Adversarial Networks (GANs) can be used to model the implicit distribution of a dataset. Traditional GANs, which only consist of a generator and a discriminator, result in networks able to generate synthetic data and distinguish real from fake samples. This adversarial game can be extended to include a classifier, which allows the training of the classification network to be enhanced with synthetic and unlabelled data. The network architecture presented in this paper is inspired by SenseGAN[1], but instead of generating and classifying sensor-recorded time series data, our approach operates with extracted features, which drastically reduces the amount of stored and processed data and enables deployment on less powerful and potentially wearable devices. We show that this technique can be used to improve the classification performance of a classifier trained to recognise locomotion modes based on recorded acceleration data ant that it reduces the amount of labelled training data necessary to achieve a similar performance compared to a baseline classifier. Specifically, our approach reached the same accuracy as the baseline classifier up to 50% faster and was able to achieve a 10% higher accuracy in the same number of epochs.

Lukas Günthermann
Sensor Technology Research Centre, University of Sussex, Brighton, United Kingdom, e-mail: l.gunthermann@sussex.ac.uk

Andrew Philippides
Centre for Computational Neuroscience and Robotics, University of Sussex, Brighton, United Kingdom, e-mail: andrewop@sussex.ac.uk

Daniel Roggen
Sensor Technology Research Centre, University of Sussex, Brighton, United Kingdom, e-mail: daniel.roggen@ieee.org

1 Introduction

The rise of smartphones and other connected devices has led to an explosive increase in sensors, which can be used to gather information about human activity patterns [2]. The data collected in this process usually includes accelerometer data and gyroscopic data. This data can be used for human activity recognition (HAR), which aims to identify actions and goals of humans based on time series data originating from a variety of on-body or ambient sensors. Cameras and microphones are also occasionally employed for vision- and audio-based activity recognition [3, 4]. The classification problem can be solved by logical reasoning [5], data mining [6], or probabilistic reasoning [7]. Most commonly though, they are addressed using an "activity recognition chain": i.e. a sequence of signal processing and classification elements primarily based on machine learning [8].

In order to obtain training data to solve the activity recognition problem, the recorded sensor inputs have to be labelled. Since observations usually occur over a long period of time, this process can be very laborious. For instance, the annotation of a dataset, which includes 25 hours of sensor data, took 7-10 hours per 30 minutes of video footage [9]. Because of this issue, several approaches to exploit unlabelled data have been explored, such as semi-supervised learning to exploit unlabelled data [10] or active learning, which attempts to prompt labels only when necessary [11]. Most recently, however, GANs have been proposed to generate synthetic data [12], which is the approach we take in this paper.

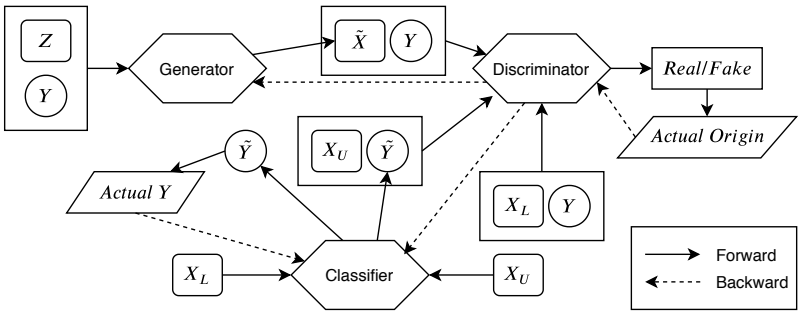


Fig. 1: The GAN architecture we use in this paper. The discriminator learns to distinguish between *real* data-label pairs originating from the training data and *fake* data, which consists of synthetic data generated by the generator with labels from the training dataset and unlabelled data with predicted labels. The discriminator serves as adversarial for the classifier, which learns how to predict labels to fool the discriminator into believing their real. Dashed backward lines represent target variables, which are compared against predicted ones to perform gradient descent. A detailed description can be found in Sec. 3.

We show that using an extended GAN framework as shown in Fig. 1 can reduce the amount of required training data for recognising human locomotion modes. The model we propose is inspired by SenseGAN[1]. It is working with extracted features instead of time-series sensor recordings, which drastically reduces the required data storage volume and simplifies the architectures of the neural networks involved. Furthermore, it is able to effectively utilise unlabelled data.

The contributions of this paper are a review of GANs and their application in activity recognition (Sec. 2), the modification of SenseGAN to operate on features (Sec. 3), the application of this architecture on the problem of recognising modes of locomotion from mobile phone sensor data (Sec. 4), an analysis of the results (Sec. 5), and discussion for future work (Sec. 6).

2 Related work

GANs, as proposed by Goodfellow [12], are a machine learning concept, which typically involves two deep neural networks competing with each other in a minimax game as shown in Eq. 1. The equation calculates the summed cross-entropy (formulated as maximisation problem) on predicting the correct origin of each data sample x[1] and $G(z)$[2] by the discriminator. The discriminator D aims to maximise this value while the generator G attempts to minimise it.

The generator takes as input noise z, which is sampled from a Gaussian normal distribution, and feeds it forward to create a synthetic sample $\tilde{x} = G(z)$ as output. These synthetic samples, together with real samples (also called training data), serve as input for the second network, the discriminator, which attempts to learn distinguishing between a real sample x and a fake sample $G(z)$. The discriminator output is defined as 1 to indicate it predicts a real sample, and 0 to predict a fake sample.

The generator is trained to attempt to fool the discriminator by creating synthetic samples, which the discriminator mistakenly classifies as real ones. In effect, the generator shapes the input distribution so that the distribution of the output samples mimics that of the training dataset. By simultaneously training both networks as adversaries, their performance can be increased until — ideally — the generator creates samples, which perfectly resemble the characteristics and diversity of the training data. In which case the discriminator can do nothing but guessing.

$$\min_G \max_D \left(E_{x\ P_{data}(x)}[log(D(x))] \ + \ E_{z\ P_z(z)}[log(1 - D(G(z)))] \right) \tag{1}$$

The loss of the discriminator is calculated based on its ability to correctly identify whether a data sample is *real* or *fake* as shown in Eq. 2. The discriminator is

[1] x is a sample originating from a given dataset.

[2] $G(z)$ is a synthetic sample originating from the generator that ought to mimic the distribution of the given dataset.

supposed to predict 1 (*real*) for a sample x and 0 (*fake*) for a synthetic sample $G(z)$. The logarithm function is used to strongly punish confident wrong predictions. As the discriminator prediction for a sample x gets closer to the correct label 1, the loss value approaches 0. If the discriminator wrongfully predicts the sample x as *fake*, the loss value will tend towards negative infinity the closer the prediction is to 0. The same goes for the outcome of predicting a label for synthetic samples $G(z)$, which is inverted by subtracting it from 1. The discriminator therefore aims to maximise this equation.

$$L^{(D)} = max[log(D(x)) \; + \; log(1 - D(G(z)))] \tag{2}$$

The loss of the generator is based on its ability to fool the discriminator into believing that its generated samples are part of the *real* data. It is estimated by presenting one of its generated synthetic samples to the discriminator and calculating the logarithm of its output value (Eq. 3), similar to the loss calculation of the discriminator. However, since the generator wants the discriminator to fail, it attempts to minimise this equation.

$$L^{(G)} = min[log(1 - D(G(z)))] \tag{3}$$

So far, GANs have been primarily used for image generation [12, 13, 14, 15, 16, 17], image transformation [18, 19, 20], or image upscaling [21]. However, there have been attempts to utilise the potential of GANs in HAR. SA-GAN [22] is able to perform knowledge transfer in the domain of wearable sensor-based HAR. GANs have also been used in action prediction on partially observable videos [23]. SensoryGAN [24] was the first framework to use GANs to generate human activity sensor data, it uses three different models, each consisting of a generator-discriminator pair generating and recognising a specific human daily activity. In contrast, SenseGAN [1] is a semi-supervised deep learning framework, which uses the same networks for multiple classes. It utilises unlabelled sensor data and improves the performance of an activity classifier by introducing it into the adversarial game between generator and discriminator, increasing the number of participating networks from two to three.

To our knowledge, no related work so far has used extracted features rather than time series as data basis for such a GAN architecture. This may lead to a significant reduction in computation time. Such a reduction would be of relevance for implementation of training (e.g. in a lifelong learning scenario) on resource-constrained devices such as wearable technology.

3 Methods

We implemented a solution inspired by SenseGAN [1], which is able to utilise unlabelled data. However, instead of processing raw sensor signals our approach works on extracted features in order to reduce the computational complexity.

3.1 GAN Architecture

The GAN is made out of three independent networks (Fig. 1): A generator G, a discriminator D, and a classifier C. The classifier predicts labels for a given data point, the generator creates synthetic samples, which resemble the original data, and the discriminator tells apart *real* from *fake* samples.

The foundation of the training process is a dataset X_{train}, which contains two subsets: X_L with corresponding one-hot labels Y and data X_U without known labels[3]. In addition, during training, additional data is created in two ways: i) A synthetic data sample $\tilde{x}$ aiming at resembling the distribution of a given class y is generated by the generator; ii) a sample x_U from the unlabelled dataset is associated with a label $\tilde{y}$ predicted by the classifier, which attempts to trick the discriminator into believing that the prediction is a real label. Only the data-label pairs (X_L,Y) are considered *real* pairs. Unlabelled data with predicted labels (X_U,$\tilde{Y}$) and data generated for given labels ($\tilde{X}$,Y) are considered *fake* pairs.

3.1.1 Concept

Data samples from X serve as input for the classifier, which predicts a corresponding label $\tilde{y}$. If the sample originates from the labelled subset X_L, the loss is calculated based on the actual label y as shown in Eq. 4, whereas t refers to one bit of the one-hot label.

$$L^{(C)} = min \sum_{i}^{8} [-t_i \cdot \log(C(x_L)_i)] \tag{4}$$

However, if the data sample is part of X_U, the data and predicted label $\tilde{y}$ are combined to serve as input to the discriminator network. In that case the higher the loss of the classifier, the better the ability of the discriminator to classify these samples as *fake* (Eq. 5).

$$L^{(C)} = min[log(1 - D(x_U, C(x_U)))] \tag{5}$$

The generator takes a random uniform distribution z and a one-hot label y as input and generates a data sample $G_{(z,y)} = \tilde{x}$, which is supposed to make the discriminator believe that it is *real*. The length of the vector z is set to 100 as it is common practice [14]. The higher loss of the generator, the better the ability of the discriminator to classify the generated samples as *fake* (Eq. 6).

$$L^{(G)} = min[log(1 - D(G(z, y), y))] \tag{6}$$

[3] In this work, X_U also originates from the SHL dataset (see Sec. 4), but we simply omitted the associated label for the purpose of experimentation.

The discriminator is a binary classifier, which takes a data-label pair as input and predicts whether it originates from the *real* dataset (x_L,y) or if it is a *fake* pair. A data-label pair is considered *fake* if either the data sample is synthetic ($\tilde{x}$,y) or the label was predicted (x_U,$\tilde{y}$) by the classifier. The loss is calculated as shown in Eq. 7.

$$\begin{aligned} L^{(D)} = max[&log(1 - D(G(z, y), y)) \\ &+ log(1 - D(x_U, C(x_U))) \\ &+ log(D(x_L, y))] \end{aligned} \tag{7}$$

The loss functions shown in Eq. 5-7 are an inverted variant of the binary cross-entropy function, turning the common minimisation problem into a maximisation one for the discriminator. Since positive and negative values in the dataset represent equal information in different directions, the sigmoid activation function was used in all but the output layers of each network. The networks were trained using the Adam optimizer [25]. We selected the learning rate α and the exponential decay rate for the first moment estimates $\beta 1$ based on the results of a grid-search. For this grid-search we tested all possible combinations of $\alpha \in [0.00015, 0.000175, 0.00020, 0.000225, 0.00025]$ and $\beta 1 \in [0.3, 0.5, 0.7, 0.9]$ on each of the three GAN networks, whereas the combinations with the highest classifier accuracy were selected. We left the exponential decay rate for the second-moment estimates at its default value $\beta 2 = 0.999$, since that is recommended for sparse gradients [25].

The architectures of the three GAN networks, i.e. the number of hidden layers, number of nodes in the hidden layers, and the shape of random noise z were established on a trial-and-error basis. Adding a second hidden layer did not benefit any of the networks, changing the shape of z to 50 or 150 did not affect the results, and the tested options for the number of nodes included 128, 256, 512, and 1024.

3.1.2 Classifier

The classifier network consists of an input layer with a size of 15 (three channels with each five features), one hidden layer with 256 nodes and an output layer with a size of 8 (representing each of the eight classes). The input and hidden nodes are activated by a sigmoid function. The one-hot label is determined by a soft-max activation function. During training the following parameters for the Adam optimizer were established: $\alpha = 0.000225$, $\beta 1 = 0.9$, $\beta 2 = 0.999$. The classifier in the GAN will from now on be referred to as *GAN-C*.

3.1.3 Generator

The generator network consists of an input layer with a size of 108 (random vector with length of 100 plus eight one-hot labels), one hidden layer with 256 nodes and an output layer with a size of 15 (a sample $\tilde{x}$ consisting of three channels with five

features each). The input and hidden nodes are activated by a sigmoid function. The output features are generated via the hyperbolic tangent function to resample the normalized data distribution between -1 and 1. During training the following parameters for the Adam optimizer were established: $\alpha = 0.00025$, $\beta 1 = 0.7$, $\beta 2 = 0.999$.

3.1.4 Discriminator

The discriminator network consists of an input layer with a size of 23 (15 features plus eight one-hot labels), one hidden layer with 256 nodes and an output layer with a size of 1 (binary representation of *real* / *fake*). The input, hidden and output nodes are activated by a sigmoid function. During training the following parameters for the Adam optimizer were established: $\alpha = 0.00025$, $\beta 1 = 0.3$, $\beta 2 = 0.999$.

3.1.5 Baseline

In addition, we include a reference classifier *Ref-C* whose purpose is to serve as baseline performance if traditional supervised training techniques, without a GAN, were used. The reference classifier is trained exclusively on labelled samples (X_L,Y), according to Eq. 4. The architecture of the reference classifier is identical to *GAN-C*.

3.2 Feature Extraction

We extract features from the dataset using a sliding window of 5s (500 samples) applied on the 3 acceleration channels. For each window we compute the mean average, standard deviation, mean-crossing-rate (mcr), kurtosis, and skewness of each of the three sensor channels. This simple set of statistical features is often employed as a first instance in activity recognition [26]. This yields a feature vector of dimension 3x5=15, which forms a single data point. We discard all windows within which multiple activities occur, typically when the user transitions from one activity to another, and windows, which contain unidentified activities (i.e. null class). All the resulting feature vectors x_i and the associated classes $c_i \in [1..8]$ form the data basis X.

3.3 Training Setup

The dataset X consists of 40,974 samples, which we divide into validation data X_{val} and training data X_{train}. In each test run, the X_{val} is made out of 50% of the available data and the remaining 50% are used to form X_{train} (20,487 samples each).

We normalize the training dataset by scaling each feature between -1 and 1, apply the same scale on the validation set, and convert the activity classes c_i into one-hot labels y_i. We then separate X_{train} further into labelled data X_L and unlabelled data X_U, where the available labels are dropped to generate the latter one. Due to the class imbalance in the dataset (Fig. 2) we oversample the minority classes in X_L using SMOTE [27]. The number of created samples differed due to the random data selection resulting in varying imbalances, usually it was about 50% of the size of X_L.

The biggest potential of the GAN-based approach lies in its ability to utilise unlabelled data and thus reduce the amount of labelled data required for an identical performance. Therefore, we used various ratios of labelled to unlabelled data to explore the impact on the training performance. The ratio of X_L to X_{train} was tested in steps of 5% from 5% up to 25%. The ratio for X_U was tested in ratios reaching from 25% to 45%, also in steps of 5%. Constellations in which the sum of labelled and unlabelled data samples exceeds the available 50%, are excluded. This results in a total of 15 different ratio constellations. If the training data does not add up to the available 50% of the dataset, the remaining data stays unused.

We run each constellation ten times over 1500 epochs of training and store the accuracy of all networks every 25 epochs.

4 Dataset

In order to evaluate our approach, as well as possible follows-up to this work, we sought a dataset, which is publicly available, offers a large amount of data, includes a wide range of different sensor modalities, and which underpins practical applications in wearable and mobile computing. We chose to employ the SHL Locomotion and Transportation dataset [28] for this purpose. This dataset is a publicly available dataset designed to evaluate methods to recognise modes of transportation and locomotions from body-worn sensors, and in particular from sensors available in modern smartphones. The recognition of modes of locomotion and transportation enables a wide range of activity- and context-aware applications [29], among others health monitoring [30], or environmental impact assessment and recommendation [31].

The complete SHL dataset comprises the recordings of all the sensors of four Mate 9 smartphones located at distinct on-body locations (hand, trouser's pocket, shirt pocket and backpack/handbag) from 3 users, who engaged in 8 different transportation and locomotion activities over a period of 7 months, in the south-east of the UK, including in London. The activities were precisely annotated from a body-worn camera, and include 8 activity classes: being still, walking, running, cycling, driving in a car, in a bus, on a train or on the subway. All the motion sensor modalities are sampled at 100Hz. Since it's preliminary release in 2017, this dataset was exhaustively analysed to understand the information content in the various data channels [32, 33] and it was used in two public machine learning challenges [34, 35].

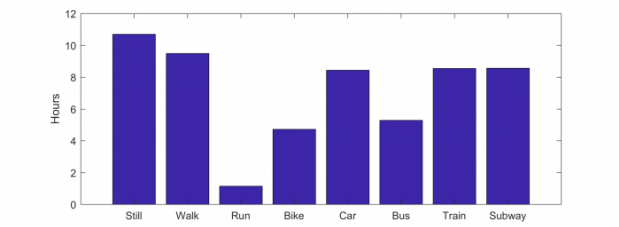


Fig. 2: Distribution of the number of recording hours according to the activities in the SHL preview dataset.

In this work, we use a subset of the dataset called the *preview version* of the dataset[4] as it was the one earliest to be released and thus allows for comparison with other published work. This preview version includes the data of 3 users, each recorded over 3 days and includes 59 hours of annotated recordings (Fig. 2 indicates the recording distribution according to the activities). In this work, we chose to use only the data of the 3D acceleration sensor from the smartphone placed in the trouser pocket of the users. A previous analysis showed that the recognition of transportation and locomotion is possible with high accuracy score even when using only the acceleration sensor data [32].

[4] http://www.shl-dataset.org/download/

Table 1: For each labelled-unlabelled ratio constellation we selected the best performance of our classifier *GAN-C* and the baseline classifier *Ref-C* together with the epoch at which they occur. The difference in accuracy displays the performance of *GAN-C* compared to *Ref-C*. A positive value indicates that *GAN-C* achieved a higher peak performance while a negative value indicates a better performance of *Ref-C*. A positive difference in epochs indicates an earlier peak of A_C while a negative value means that *Ref-C* peaked before *GAN-C*. Furthermore, for each constellation, we estimated the epoch when *GAN-C* had the biggest advantage compared to *Ref-C*, i.e. the epoch at which *GAN-C* achieved an accuracy, which *Ref-C* reached way later. A peak advantage of 50 at epoch 75 means that at epoch 75 our classifier provided a recognition accuracy for which to reach the reference classifier needed 50 more epochs of training.

Ratio X_L	5%	5%	5%	5%	5%	10%	10%	10%	10%	15%	15%	15%	20%	20%	25%
Ratio X_U	25%	30%	35%	40%	45%	25%	30%	35%	40%	25%	30%	35%	25%	30%	25%
Peak A_C	0.8685	0.863	0.8583	0.8531	0.8297	0.8696	0.865	0.8617	0.8519	0.8704	0.8589	0.8566	0.8734	0.8686	0.8672
Epoch	375	475	550	775	925	350	550	675	625	375	350	525	425	600	425
Peak R_C	0.8741	0.8692	0.8675	0.8488	0.8314	0.8769	0.8718	0.861	0.8545	0.8704	0.8654	0.8666	0.8743	0.8685	0.8616
Epoch	550	850	1050	900	1400	700	750	975	1075	475	625	1075	575	700	325
Diff. Acc.	-0.0064	-0.0071	-0.0107	0.0051	-0.0021	-0.0084	-0.0078	0.0008	-0.0031	0	-0.0075	-0.0115	-0.001	0.0001	0.0064
Diff. Epoch	175	375	500	125	475	350	200	300	450	100	275	550	150	100	-100
Peak Ad.	50	100	125	450	225	75	100	300	225	125	100	75	125	150	50
Epoch	75	250	350	450	500	250	300	675	425	350	275	150	375	350	150

5 Results

The development of classification accuracy over the training epochs can be seen in Fig. 3a. We calculated the accuracy A_C for our classifier *GAN-C* and A_R for the reference classifier *Ref-C* by evaluating them on the validation dataset. The accuracy A_D reflects the ratio of samples the discriminator labelled correctly, when using the validation dataset X_{Val} and an equal number of synthetic samples $\tilde{X}$ as input. The generator's accuracy A_G represents the ratio of synthetic samples, which the discriminator misclassified as *real*. The performance P_C represents the accuracy of *GAN-C* relative to the accuracy of *Ref-C*, calculated by dividing A_C by A_R. A value bigger than 1 means that *GAN-C* achieved a higher accuracy and values below 1 indicate that *Ref-C* performs better than *GAN-C*. For all accuracies we used the mean average of all ten runs for each labelled-unlabelled ratio constellations.

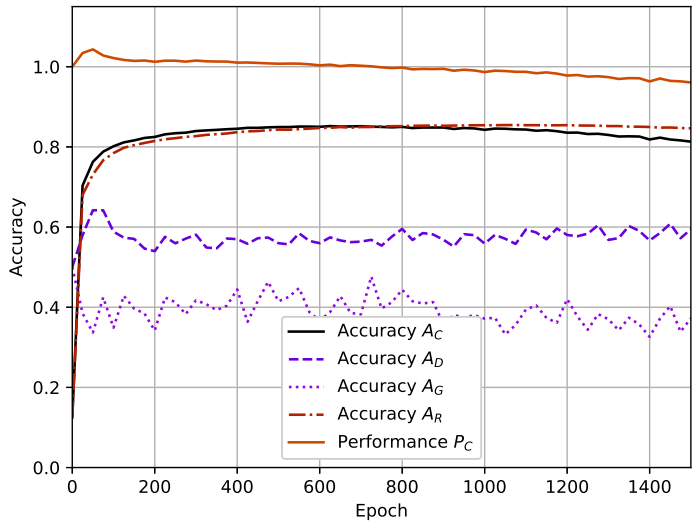
The results show a lot of similarity between the classification accuracy of our classifier and the reference classifier. It can be seen that the GAN-based approach usually leads to a slight increase in accuracy in the initial epochs, but a drop as the number of epochs increases. In contrast, the reference classifier keeps a constant level also with higher number of epochs.

Figure 4a shows the results after 50 epochs of training, which suggest that the accuracy for class 1 is responsible for the later drop in accuracy. The other major source for error is the indistinguishability between class 7 and 8 due to the similarity of train and subway movements. All three of these classes are activities where the user tends to be still in relation to their surroundings.

Table 2: The peak performance of our classifier compared to the baseline classifier over the whole run time. The performance P_C is calculated by dividing A_C by A_R. For representation, the performance is portrait as $P_C - 1$ which equals dividing the difference in accuracy between *GAN-C* and *Ref-C* by the accuracy of *Ref-C*. The maximum value across all evaluated epochs is selected as peak.

Data Ratios		Ratio X_L				
		5%	10%	15%	20%	25%
Ratio X_U	25%	2.73%	2.57%	3.12%	3.20%	3.15%
	30%	3.76%	3.39%	3.48%	3.94%	-
	35%	3.78%	3.89%	4.08%	-	-
	40%	3.37%	4.33%	-	-	-
	45%	10.28%	-	-	-	-

We evaluated the data stored every 25 epochs and calculated the performance ratio P_C. Table 2 shows that the increase in performance correlates with an increasing ratio of unlabelled data. It is worth mentioning that all of these peaks occurred after



(a) Development of accuracy over epochs

(b) Final confusion matrix

Fig. 3: Results with 10% labelled and 40% unlabelled data after 1500 epochs of training. a) shows the accuracy over the whole training process, whereas A_C represents the accuracy of the GAN classifier and A_R the accuracy of the baseline classifier. The performance P_C shows the relation between these two classifiers, values above 1 indicate a superior GAN classifier while 0 indicates a superior baseline classifier. A_D is the accuracy of the discriminator and A_G shows the accuracy of the generator. b) shows the final confusion matrix for the GAN classifier.

25 epochs of training, with the exception of the ratio constellation 10% labelled and 40% unlabelled, which peaked after 50 epochs. As expected, the performance peak increases the more unlabelled data is available, indicating that *GAN-C* was able to utilise unlabelled data.

The highest achieved classification accuracy over the whole of each run can be found in Table 1. Although *Ref-C* overall achieves slightly higher accuracies on average, *GAN-C* reaches these peaks significantly earlier.

Furthermore we calculated the horizontal difference by comparing the number of epochs one classifier required to reach the accuracy of the other. This represents the advantage in terms of the largest reduction in the number of training epochs required by *GAN-C*, where both *GAN-C* and *Ref-C* have identical performance. Again, we estimated the peak values for *GAN-C*, which are shown in Table 1. Compared to the peak P_C values, these ones are spread all across the first half of training epochs.

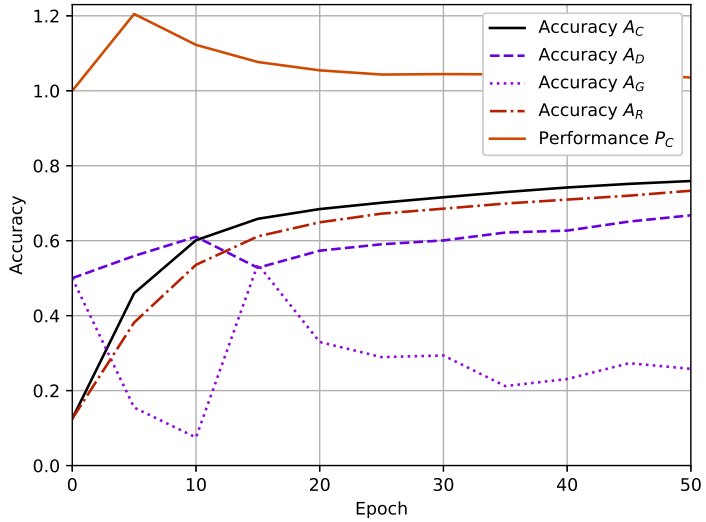
6 Discussion

The results clearly show that the GAN-based approach is able to improve the accuracy of a classifier in locomotion recognition tasks, or to reduce the number of training epochs for an identical performance compared to a traditional classifier.

However, there are downsides, such as the rapid decrease in accuracy after many epochs of training. We suspect that one of the reason for the decrease in accuracy is due to overfitting, a rather common and not GAN-specific issue [36]. It must also be considered that the GAN architecture is not optimised, e.g. the grid-search for learning and decay rate could be greatly extended. Other hyper-parameters, such as the number of hidden layers or the number of nodes in the hidden layers were established on a trial-and-error basis. Any future approach should consider optimising these parameters as well, via, for instance, random search [37], by a sequential algorithm [38], or by an evolutionary algorithm [39].

For the scope of this paper, we only used 5 selected features of the available sensor data in order to keep the required computational power as low as possible. Therefore, future approaches should definitely consider using more features, as previous work using the SHL dataset explored up to 2727 features[32]. This would enable the investigation of the trade-off between a reduction in computational complexity and the potential of saving resources due to early convergence of more complex classification models. An approach using raw data could serve as a benchmark for the potential of SenseGAN [1].

When preparing the dataset, we took all captured windows and shuffled them before separating them into training and validation data. We did this to solely explore the application of a GAN, and did not want to take into consideration here applicative aspects, in order to isolate the characterisation of the GAN from other aspects. By applicative aspect, we mean that devising a system, which could be "user invariant" would have needed leave-one-user-out crossvalidation or a system suitable for long-term stability would have required a leave-one-time-period-out crossvalidation. By



(a) Development of accuracy over epochs

(b) Final confusion matrix

Fig. 4: Results with 10% labelled and 40% unlabelled data after 50 epochs of training.

shuffling the data, we ignore this, and therefore cannot make claims to the performance if the system would be applied in the field. Furthermore, since both, training and validation data, stem from the same sensor modalities, we applied the scaling range established in the training data set also to the validation data set. In a practical application, this would only be possible if the sensor signals, which are classified, originate from the same source or an identical setup than the one the network was trained on, otherwise the new data would have to be scaled based on its own range.

Another attempt worth trying would be implementing individual generator networks for each output class, as done in SensoryGAN [24]. As shown in the confusion matrices in the result section, it is most often class 1 with which the classifiers struggle, having an independent network just for that class might reduce this issue.

At the moment the performance of the GAN is measured entirely by the accuracy of the classifier, however methods to calculate the distance between generated and original samples could be used to find suited parameters for the generator, which should improve the classifier in the long run as well.

7 Conclusion

We have shown that a GAN framework, which includes a classifier and works on extracted features, can be used to improve the classification of locomotion modes. This reduces the computational complexity compared to classifiers operating on raw data and potentially enables the implementation on wearable devices for continual lifelong learning. The results have shown that our classifier at its peak was able to achieve the same accuracy with only half as many training epochs as the baseline classifier. In another setting it was able to achieve a 10% higher accuracy in the same number of epochs. Future approaches should explore if adding more extracted features from the dataset can have a positive impact on the classifiers ability while still allowing deployment to devices with reduced computational power.

Acknowledgements We acknowledge NVIDIA for their donation of a TITAN XP.

References

1. Yao, S., Abdelzaher, T., Zhao, Y., Shao, H., Zhang, C., Zhang, A., Hu, S., Liu, D., Liu, S., Su, L., et al.: Sensegan: Enabling deep learning for internet of things with a semi-supervised framework. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies **2**(3) (2018) 1–21
2. Vaizman, Y., Ellis, K., Lanckriet, G.: Recognizing detailed human context in the wild from smartphones and smartwatches. IEEE Pervasive Computing **16**(4) (2017) 62–74
3. Richoz, S., Birch, P., Ciliberto, M., Wang, L., Gjoreski, H., Perez-Uribe, A., Roggen, D.: Human and machine recognition of transportation modes from body-worn camera images. In: Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), IEEE (2019)

4. Ward, J.A., Lukowicz, P., Troster, G., Starner, T.E.: Activity recognition of assembly tasks using body-worn microphones and accelerometers. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(10) (2006) 1553–1567
5. Kautz, H.: A formal theory of plan recognition. PhD thesis, University of Rochester (1987)
6. Tao Gu, Zhanqing Wu, Xianping Tao, Pung, H.K., Jian Lu: epsicar: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition. In: 2009 IEEE International Conference on Pervasive Computing and Communications. (2009) 1–9
7. Charniak, E., Goldman, R.P.: A bayesian model of plan recognition. Artificial Intelligence **64**(1) (1993) 53 – 79
8. Bulling, A., Blanke, U., Schiele, B.: A tutorial on human activity recognition using body-worn inertial sensors. ACM Computing Surveys **46**(3) (2014)
9. Roggen, D., Calatroni, A., Rossi, M., Holleczek, T., Förster, K., Tröster, G., Lukowicz, P., Bannach, D., Pirkl, G., Ferscha, A., Doppler, J., Holzmann, C., Kurz, M., Holl, G., Chavarriaga, R., Sagha, H., Bayati, H., Creatura, M., d. R. Millán, J.: Collecting complex activity datasets in highly rich networked sensor environments. In: 2010 Seventh International Conference on Networked Sensing Systems (INSS). (2010) 233–240
10. Miu, T., Missier, P., Plötz, T.: Bootstrapping personalised human activity recognition models using online active learning. In: 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing. (2015) 1138–1147
11. Zeng, M., Yu, T., Wang, X., Nguyen, L.T., Mengshoel, O.J., Lane, I.: Semi-supervised convolutional neural networks for human activity recognition. In: 2017 IEEE International Conference on Big Data (Big Data). (2017) 522–529
12. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. arXiv preprint (2014) arXiv:1406.2661
13. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X., Chen, X.: Improved techniques for training gans. In Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R., eds.: Advances in Neural Information Processing Systems 29. Curran Associates, Inc. (2016) 2234–2242
14. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint (2015)
15. Jolicoeur-Martineau, A.: The relativistic discriminator: a key element missing from standard gan. arXiv preprint (2018) arXiv:1807.00734
16. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive Growing of GANs for Improved Quality, Stability, and Variation. arXiv e-prints (2017) arXiv:1710.10196
17. Brock, A., Donahue, J., Simonyan, K.: Large Scale GAN Training for High Fidelity Natural Image Synthesis. arXiv e-prints (2018) arXiv:1809.11096
18. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-Image Translation with Conditional Adversarial Networks. arXiv e-prints (2016) arXiv:1611.07004
19. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. arXiv e-prints (2017) arXiv:1711.11585
20. Wu, H., Zheng, S., Zhang, J., Huang, K.: GP-GAN: Towards Realistic High-Resolution Image Blending. arXiv e-prints (2017) arXiv:1703.07195
21. Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., Shi, W.: Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. arXiv e-prints (2016) arXiv:1609.04802
22. Soleimani, E., Nazerfard, E.: Cross-Subject Transfer Learning in Human Activity Recognition Systems using Generative Adversarial Networks. arXiv e-prints (2019) arXiv:1903.12489
23. Wang, D., Yuan, Y., Wang, Q.: Early action prediction with generative adversarial networks. IEEE Access **7** (2019) 35795–35804
24. Wang, J., Chen, Y., Gu, Y., Xiao, Y., Pan, H.: SensoryGANs: An effective generative adversarial framework for sensor-based human activity recognition. In: 2018 International Joint Conference on Neural Networks (IJCNN). (2018) 1–8

25. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014)
26. Figo, D., Diniz, P.C., Ferreira, D.R., Cardoso, J.M.P.: Preprocessing techniques for context recognition from accelerometer data. Personal and Ubiquitous Computing **14**(7) (2010) 645–662
27. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: Synthetic minority oversampling technique. J. Artif. Int. Res. **16**(1) (2002) 321–357
28. Gjoreski, H., Ciliberto, M., Wang, L., Ordonez Morales, F.J., Mekki, S., Valentin, S., Roggen, D.: The University of Sussex-Huawei Locomotion and Transportation Dataset for Multimodal Analytics With Mobile Devices. IEEE Access **6** (2018) 42592–42604
29. Engelbrecht, J., Booysen, M.J., van Rooyen, G., Bruwer, F.J.: Survey of smartphone-based sensing in vehicles for intelligent transportation system applications. IET Intelligent Transport Sys. **9**(10) (2015) 924–935
30. Gjoreski, H., Kaluza, B., Gams, M., Milic, R., Lustrek, M.: Context-based ensemble method for human energy expenditure estimation. Appl. Soft Comput. **37** (2015) 96–970
31. Anagnostopoulou, E., Urbancic, J., Bothos, E., Magoutas, B., Bradesko, L., Schrammel, J., Mentzas, G.: From mobility patterns to behavioural change: leveraging travel behaviour and personality profiles to nudge for sustainable transportation. J. Intelligent Information Sys. **2018** (2018) 1–22
32. Wang, L., Gjoreski, H., Ciliberto, M., Mekki, S., Valentin, S., Roggen, D.: Enabling reproducible research in sensor-based transportation mode recognition with the Sussex-Huawei dataset. IEEE Access **7** (2019) 10870–10891
33. Wang, L., Gjoreski, H., Ciliberto, M., Mekki, S., Valentin, S., Roggen, D.: Benchmarking the SHL recognition challenge with classical and deep-learning pipelines. In: Proc. ACM Int Joint Conf and 2018 Int Symp on Pervasive and Ubiquitous Computing and Wearable Computers, ACM (2018) 1626–1635
34. Wang, L., Gjoreski, H., Murao, K., Okita, T., Roggen, D.: Summary of the Sussex-Huawei Locomotion-Transportation Recognition Challenge. In: Proc. ACM Int Joint Conf and 2018 Int Symp on Pervasive and Ubiquitous Computing and Wearable Computers, ACM (2018) 1521–1530
35. Wang, L., Gjoreski, H., Ciliberto, M., Lago, P., Murao, K., Okita, T., Roggen, D.: Summary of the Sussex-Huawei locomotion-transportation recognition challenge 2019. In: Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers, ACM (2019) 849–856
36. Hawkins, D.: The problem of overfitting. Journal of chemical information and computer sciences **44** (2004) 1–12
37. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of Machine Learning Research **13**(10) (2012) 281–305
38. Bergstra, J.S., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In Shawe-Taylor, J., Zemel, R.S., Bartlett, P.L., Pereira, F., Weinberger, K.Q., eds.: Advances in Neural Information Processing Systems 24. Curran Associates, Inc. (2011) 2546–2554
39. Stang, M., Meier, C., Rau, V., Sax, E.: An evolutionary approach to hyper-parameter optimization of neural networks. In Ahram, T., Taiar, R., Colson, S., Choplin, A., eds.: Human Interaction and Emerging Technologies, Cham, Springer International Publishing (2020) 713–718