Generating a Personalised Sensor Data Generation System for the Fusion of Adversarial Networks and Behavioural Matter-of-Fact Mapping

Jingge Cao[1,2], Lijuan Sun[1,3(✉)], Jingchen Wu[1], Xiujian Zhang[3], and Xu Wu[1,2]

[1] Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, Beijing, China
{caojingge,sunlijuan,wux}@bupt.edu.cn
[2] School of Cyberspace Security, BUPT, Beijing, China
[3] Key Laboratory of Artificial Intelligence Measurement and Standards for State Market Regulation, Beijing Aerospace Institute for Metrology and Measurement Technology, Beijing 100076, China

Abstract. Generative models such as Generative Adversarial Networks and Variational Self-Encoders have been shown to generate highly simulated synthetic data with incredible results in areas such as text, images, and audio. However, to date there is no proven method for generating sensor data for mobile terminals. This is because sensor-generated data is high-dimensional, high-complexity, and contains a lot of noise and variations, which makes it difficult for generative models to learn realistic distributions from this data. In addition, different users acquiring sensor data for the same action will also have different features, which makes it more challenging to generate fine-grained sensor data. In order to solve the above problems, we propose a personalised sensor data generation system for fusion of generative adversarial networks and behavioural matter-of-fact mapping, which creatively introduces matter-of-fact mapping and aims to reduce the probability of occurrence of situations that do not conform to the matter-of-fact logic in the personalised sensor data set by utilising matter-of-fact logic and temporal relationships originated from matter-of-fact mapping, and thus improve the accuracy of user behavioural sensor data generation. The results demonstrate that the sensor data generated by the method proposed in this paper achieves 90% accuracy under the behavioural action recognition task and generates personalised sensor data for a long period of time through the designed state transfer template.

Keywords: generative adversarial networks · variational self-encoder · Markov chain · matter-of-fact mapping

J. Cao and L. Sun—Contributed equally to this work.

W. Yu and L. Xuan (Eds.): MLICOM 2023, LNICST 518, pp. 81–95, 2024.
https://doi.org/10.1007/978-3-031-71716-1_7

1 Introduction

With the arrival of the mobile Internet era, major applications can additionally acquire various types of sensor data on mobile terminals, such as GPS information, gyroscope data, etc., and then build a virtual user portrait by combining the user's access to social platforms. The data collected by sensors is crucial for data analysis and applications. When performing big data analysis, we must consider the privacy of user sensor data. To address this issue, in this paper, we propose a personalised sensor data generation system that fuses generative adversarial networks and behavioural matter-of-fact mapping to generate realistic sensor data instead of real user-sensitive data, based on the research of Alzantot et al. [6]. We combine Markov chains with matter-of-fact mapping to associate user roles and behavioural keywords with different potential relationships, which helps reduce the probability of occurrence of illogical situations in the personalised sensor data set while providing more interpretability for user personalised sensor data generation. In addition, we design a data generation model using the idea of generative adversarial networks, which can generate virtual sensor data and enables the generated sensor data to not only protect user privacy, but also maintain the same statistical properties as the real data, thus ensuring the accuracy and reliability of data analysis.

Our contribution points are summarised as follows:

1) An improved Markov chain role sensor behavioural feature state transfer algorithm based on matter-of-fact logic constraints is designed to reduce the probability of occurrence of non-matter-of-fact logic situations in personalised sensor datasets.
2) The structure of generative adversarial network is improved, the variational self-encoder is used as the generator of the model, a high accuracy multi-classifier is designed as the discriminator of the model, and the accuracy of the generated three-axis sensor data reaches 90% under the action classification task.
3) By combining with matter-of-fact mapping and state transfer templates, a method for generating personalised behavioural feature sensor data for virtual identity is proposed, which can generate personalised sensor sets that are long-lasting, continuous, and conform to the logic of matter-of-fact mapping.

2 Related Work

In recent years, along with the research breakthroughs of data generation models in the field of time series synthesis, how to apply generative adversarial networks and their derived models to the field of sensor data generation has received more and more attention from researchers. At present, some progress has been made in the research of sensor data generation. In 2017, M. Alzantot et al. proposed a model named SenseGen [4], which for the first time draws on the idea of generative adversarial to generate sensor data, except that the generator and the discriminator of the model are trained separately, so SenseGen does not belong to generative adversarial networks in nature, and the quality of the data generated by SenseGen was not satisfactory. In the same year, in order to generate high-quality medical time series data, L. Hyland et al. proposed the model RGAN that can generate real data [3]. The model follows the architecture of generative adversarial networks, with RNN networks for both generator and discriminator, and has achieved good

results on the ICU dataset. However, this model cannot avoid the defect of traditional generative adversarial networks, i.e., the "pattern collapse" problem. The data generated by RGANs are almost completely similar with low diversity. In 2018, J. Wang et al. proposed a model called SensoryGANs [5]. The model designed three different model structures for three different actions: standing still, walking, and jogging. SensoryGANs are the first complete generative adversarial networks for generating sensor data applied to the research field of HAR (Human Behaviour Recognition). However, the structure of the model design is still rudimentary and is not a unifying model for multiple behavioural actions. The three actions generated by SensoryGANs are coarse-grained data and the model does not generate jogging actions well. In 2022, Moustafa Alzantot et al. proposed the PhysioGAN model [6]. This model combines a variational self-encoder with an adversarial network for the task of conditional generation of synthetic sensor data by using a novel training method that combines a variational self-encoder and a generative adversarial network. The PhysioGAN model generates data that is 10% to 20% less accurate than the real data for classification tasks on the HAR and AFib datasets, making a significant breakthrough. In addition PhysioGAN introduces latent spatial noise to avoid homogenisation of the generated data. However, the data generated by PhysioGAN has a concentration of 60% to 80% accuracy on classification models, and its performance is not perfectly adapted to the needs of high accuracy.

However, compared to the great successes harvested in the fields of image generation and audio generation, much less progress has been made in generating high-quality personalised sensor datasets, and there is still a lot of room for research on generating sensor data through generative adversarial networks and variational self-encoders. The scheme proposed in this paper introduces the matter-of-fact map as auxiliary information, generates state transfer templates with the help of matter-of-fact map through Markov chains, and then splices the generated individual action sensor data into the user's one-day behavioural data through the state transfer templates, which in turn can better enhance the accuracy of the generation of personalized user behavioural sensor data.

3 Algorithm Model

3.1 Overall Structure

The core structure of the system designed in this paper is a generative adversarial network, which contains a generator and a discriminator, where the generator consists of a variational self-encoder. The overall structure of the designed model is shown in Fig. 1. The discriminator is trained to distinguish between the samples generated by the generator when fed with random noise and the samples extracted from the real dataset, and provides feedback to continuously improve the quality of the generated data, which in turn generates single action data with high accuracy. These data are then used to generate state transfer templates for the different actors by referring to the Markov chain of the matter-of-fact mapping to synthesise personalised sensor data in terms of the degree of day.

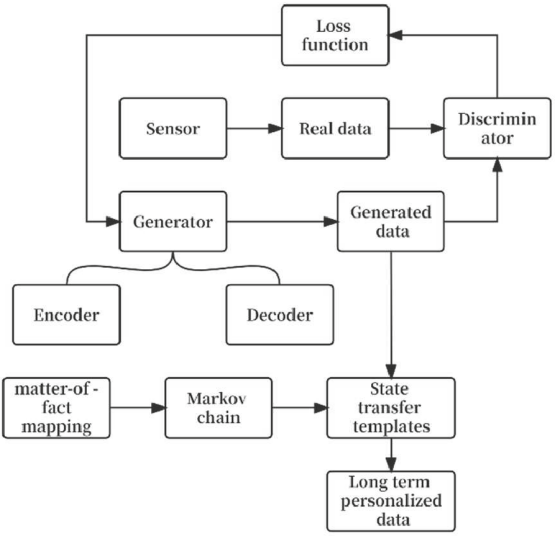

Fig. 1. Overall Structure

3.2 Generation of State Transfer Templates

In order to generate user personalised behavioural sensor data in terms of days, state transfer templates corresponding to roles and labels need to be generated using matter-of-fact mapping and Markov chains.

The construction of the matter-of-fact map of the user personalised sensor domain includes schema layer construction and data layer construction. The schema layer is an organisational framework for describing entity concepts, attributes, and inter-entity relationships. The data layer construction is the selection of appropriate methods for entity and relationship extraction of user personalised sensor domain knowledge. The entities and relationships in the data layer are associated and mapped according to the preset method in the schema layer, so as to obtain the user personalised sensor domain factual map. The model layer and data layer are constructed using top-down and bottom-up approaches, respectively. The schema layer is constructed by extracting the corresponding entities and attributes from the statistically obtained semi-structured and unstructured data of user personalised sensors, including three layers of entity nodes, namely, roles, sticky notes and keywords, so as to realise the construction of the matter-of-fact graph. The matter-graph can be represented by $G = (E, R, S)$. Where $E = \{e_1, e_2, \ldots\}$ denotes the set of entities in the knowledge base, $R = \{s_1, s_2, \ldots\}$ denotes the set of relationships, and S denotes the "node-relationship-node" triad knowledge set. The data layer construction is guided by the schema layer organisational framework to extract the required entities and relationships from the statistically captured text including roles, tags and keywords.

Markov chain, as a special kind of probabilistic model, is the study of the state and state transfer of an operating system. The human activity process can be abstracted as a state transfer process, so this paper chooses to use the Markov chain method to study the

state changes of a person's daily activities and to construct a state transfer matrix using applied statistical methods.

Suppose the prediction object has $X_i(i = 1,2,\ldots\ldots, n)$ states, in the collected data, the total amount of data that is in state X_i is a_i, and the amount of data that is transferred from state X_i to X_j is a_{ij}, which satisfies $\sum_{j=1}^{n} a_{ij} = a_i(i = 1,2,\ldots\ldots, n)$. Then the transfer frequency from state X_i to X_j is $f_{ij} = \frac{a_{ij}}{a_i}$ $(i = 1,2,\ldots\ldots, n)$. From the knowledge of probability theory, when the theoretical distribution of state probability is unknown, if the sample capacity is large enough, the theoretical distribution of the state can be approximated by the sample distribution. Therefore, for the unknown transfer probability, the transfer frequency can be used to approximate the transfer probability. Therefore, the estimate of the transfer probability from state X_i to X_j is $\hat{p}_{ij} \approx \frac{a_{ij}}{a_i}$. After data collection, the probabilities of transferring each state of a character to each other are tabulated, and an estimate of the one-step transfer probability matrix is obtained, as shown in Eq. 1:

$$\hat{P} = \begin{bmatrix} \hat{p}_{11}\ \hat{p}_{12}\ \cdots\ \hat{p}_{1n} \\ \hat{p}_{21}\ \hat{p}_{22}\ \cdots\ \hat{p}_{2n} \\ \cdots\ \cdots\ \cdots\ \cdots \\ \hat{p}_{n1}\ \hat{p}_{n2}\ \cdots\ \hat{p}_{nn} \end{bmatrix} \tag{1}$$

where row i represents the transfer probability vector of the behavioural state X_i. The Markov chain prediction model formula is shown in Eq. 2:

$$\mu_{n+1} = \mu_n P \tag{2}$$

where μ_n is the probability vector of the predicted object at time period n, P is the transfer probability matrix, and μ_{n+1} is the probability vector of the predicted object at time period $n + 1$, i.e., the predicted outcome. Since $\mu_2 = \mu_1 P$, Eq. 3 can also be derived:

$$\mu_{n+1} = \mu_1 P_{n+1} \tag{3}$$

where μ_1 is the initial state vector of the predicted object and P is the transfer probability matrix. Using Eqs. 2 or 3 the prediction of the state can be carried out to get the state transfer process of a character for one day.

3.3 Overall Structure of Generative Model

Generative Adversarial Network and Variational Self-Encoder have been proved to be able to generate multimodal data effectively, however, the research on the generation of three-axis sensor data for mobile terminals is still relatively little, this paper utilises an improved model combining the Generative Adversarial Network and Variational Self-Encoder, which reduces the likelihood of the Generative Adversarial Network modal collapse and modal collapse problems, and thus improves the overall stability of the model. The variational self-encoder can map data from the high-dimensional space to

the potential space, while the generative adversarial network can generate data from the potential space, and this combination can improve the ability of data dimensionality reduction. In order to further improve the accuracy and realism of the data generated by the model, in this paper, we design a high-accuracy deep learning classification model for multiple behavioural actions and use it as a discriminator for generative adversarial networks for model training. The overall architecture of our proposed data generation model shown in Fig. 2 mainly consists of a variational self-encoder module and a discriminator module, which in turn consists of an Encoder and a Decoder.

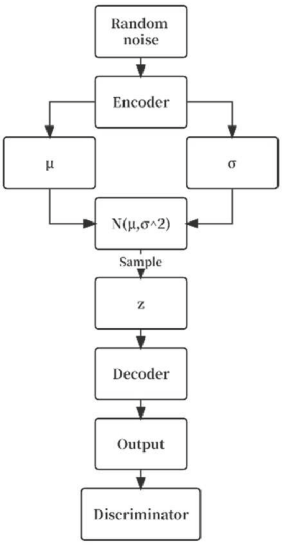

Fig. 2. Generative Adversarial Network Structure

3.4 Discriminator Structure

The auxiliary classifier model designed in this paper contains four one-dimensional convolutional layers, each of which uses 64 filters, with the kernel_size set to 3, the activation function is the ReLU function, and the padding method is same. Then a Flatten layer is used to spread the output of the convolutional layers, followed by a dropout layer and a BatchNormalisation layer. The dropout layer is used to reduce overfitting and the BatchNormalisation layer is used to speed up the training process, making the model more robust and better adapted to different data. Two fully connected layers are used at the end of the model, where the first fully connected layer has 256 nodes and the activation function is ReLU, while the number of nodes in the second fully connected layer is determined by the actual number of categories in the dataset, and the activation function uses softmax.

The structure of the discriminator is basically the same as the classifier, the only difference is that the classification result of the discriminator is $n + 1$ classes, where n

is the number of actions to be classified, and the extra classes are called "false" classes or "generated" classes. The purpose of this design is to allow the discriminator to have more information to learn, so as to improve its accuracy in distinguishing between true and false data, which in turn improves the training effect of the generative adversarial network.

3.5 Variational Self-encoder Structure

The generator designed in this paper defines a variational self-encoder class, which consists of an RNN encoder and an RNN decoder. The RNN encoder encodes the input x as a latent space vector z, and also returns the mean and variance of z, which is used to compute the KL dispersion. The RNN decoder accepts the latent space vector z and the label y as inputs, and generates the reconstructed output. Finally, the class method returns the reconstructed output, mean and variance.

The structure of the RNN encoder is shown in Fig. 3. The main role of the RNN encoder is to encode the input sequence x into a fixed-length vector z, and to compute the mean and variance of z. Specifically, the encoder consists of the following components.

(1) Initialisation method: the method defines some hyperparameters (e.g., number of rnn units, feedback direction), as well as a GRU layer and two fully connected layers for finding the mean and variance, respectively.
(2) Reparameterisation method: the reparameterisation trick is implemented. The trick converts the Gaussian distributed random noise ε into a new coded vector z by learning the mean μ and standard deviation parameter σ, i.e.

$$z = \mu + \varepsilon * \sigma, \ \ \sigma \in N(0, 1)$$

In this way decoupling and microscopicity can be maintained and it helps to improve training stability.
(3) Core method: it receives an input tensor (3D tensor) and returns the coding vector z, the mean vector and the variance vector. In the execution process, it first uses the GRU layer to process the input x, then extracts the output h of the last time step, then obtains the mean and variance vectors through the fully connected layer, respectively, and finally calls the reparameterisation method to obtain the final coding vector z.

The structure of the RNN decoder its main role is to decode a fixed-length vector z and a sequence of labels y into a sequence x of length max_len of the time step.

(1) Initialisation method: the method defines some hyperparameters (e.g., number of rnn units, number of features, number of labels), as well as three GRU layers and a fully connected layer.
(2) Core method: it receives the input vector x, the hidden state hidden and the label sequence y of the current time step, and returns the output vector output and the updated hidden state new_hidden of the current time step. In the execution process, it firstly performs the word embedding operation on the label sequence y, and then splices the embedding vector yemb with the input vector x by the last dimension to obtain the new input tensor rnn_input. Next, the three GRU layers are utilised to process rnn_input and the hidden state of the previous layer respectively, and the

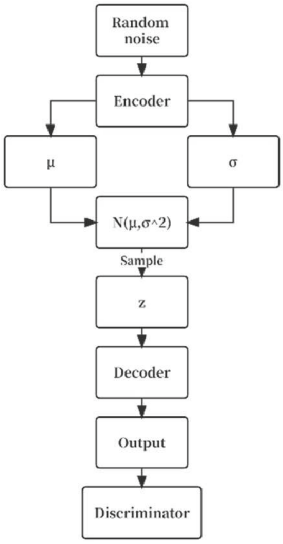

Fig. 3. RNN Encoder structure

output of the last layer is mapped to the desired output vector dimension through the fully connected layer. Finally, the hidden states of each time step are sequentially arranged to obtain the new hidden state new_hidden. Note that since the GRU layer is a stateful network layer, the hidden state of the previous moment needs to be used as the input of the current moment, which is why the hidden parameter needs to be passed in the method.

The training process of the model is elaborated next:

Firstly, the evaluate function of the auxiliary classifier evaluates the sample condition labels to calculate the sampling accuracy of the generative model g_model and the auxiliary model aux_model under the condition that the maximum sequence length is max_len, and the result is stored in the variable sampling_acc. Then, the generative model g_model is used to sample the fixed sampling labels and the fixed noise vector sampling_z to obtain the generative samples with the maximum sequence length of max_len, and the results are stored in the variable test_samples. Finally, the various parameters and models in the training process are saved to a file for subsequent use. Among them, the various parameters that need to be adjusted include the batch size of the training data batch_size, the number of training rounds num_epoch, the hidden space noise dimension z_dim, the number of pre-training rounds pre_train_epoch, the number of RNN units num_units, and the learning rate learning_rate.

During the training process, each batch in the training data is iterated by using a loop. For each batch, the predicted value of the model, the reconstruction loss and the KL scatter loss are calculated using the gradient bands and the total loss is calculated using these values. The gradient was then calculated and applied to the trainable variables of the model. The average values of the reconstruction loss and KL scatter loss are also tracked during training and they are updated at the end of each batch. Finally, the method

returns the average of the reconstruction loss and the KL scatter loss. We also introduced an inverse sigmoid decay method during training as a way to gradually adjust the weight share of the reconstruction loss and KL scatter loss in the total loss. In the pre-training stage, we increase the weight of the reconstruction loss so that the model increases its ability to learn the details of the original data; in the late stage of training, we increase the weight of the KL scatter loss so that the model increases the intraclass variance of the data and improves the diversity of the generation. As shown in Fig. 4, the change curve of weights conforms to the inverse sigmoid function, which reflects the idea of model confrontation.

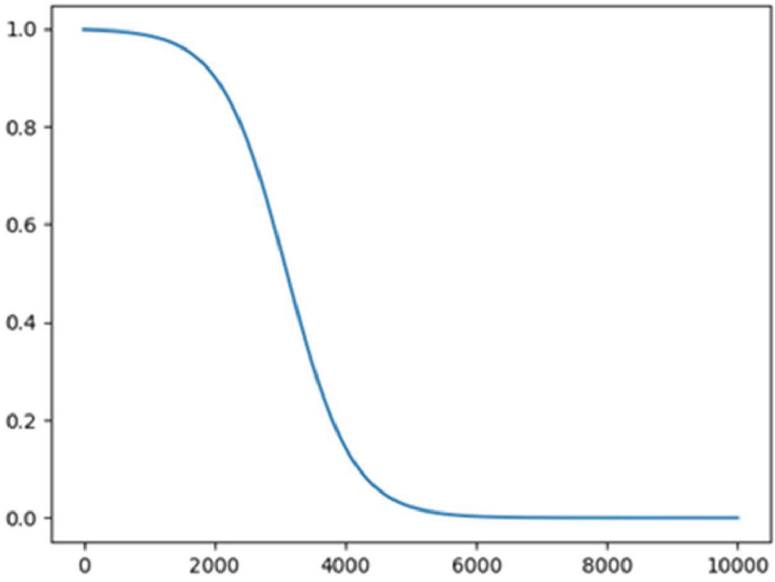

Fig. 4. Weight change curve

4 Experiments

In order to assess the effectiveness of the model, we consider and evaluate the following three main aspects. Firstly, visual assessment, the sensor data images generated by the generator should look realistic in appearance (e.g., the sensor data have similar maxima, similar ranges of values, etc.), so much so that it is difficult for a third party to distinguish them from the real data sampled in the training samples by observation. Second, the generated data should also mimic the same salient features as the real data. Thus, any analytical function computed on synthetic data should return a return value close to the same function computed on real data. Finally, the model should have the ability to generate personalised data, i.e., to avoid generating the same type of action data similar to that of different users and being detected as false user data. Therefore, this paper

proposes to measure the degree of similarity between data by calculating the DTW distance (Dynamic Time Warping distance) of different data under the same class of actions.

In addition to this, the experiment also demonstrates the state transfer templates generated based on the matter-of-fact mapping and Markov chain, and the personalised behavioural profile sensor data of different roles are generated based on the templates.

4.1 Visual Assessment

The visual evaluation mainly contains two aspects, one is the comparison between the generated data and the real data of the same action, and the other is the comparison between the generated data of different actions. As shown in Fig. 5, the left figure shows the real data of the walking action, and the right figure shows the data generated by the model. The three sub-figures are, from top to bottom, the acceleration sensor waveform, the angular velocity sensor waveform and the orientation sensor waveform, and the blue, red and yellow colours represent the data in the *x*-axis, *y*-axis, and *z*-axis. Visually, it can be seen that the generated sensor data conforms to the general pattern of the real data, and the characteristics of the data mean value, maximum value, minimum value, and range of values are also approximately the same as the real data.

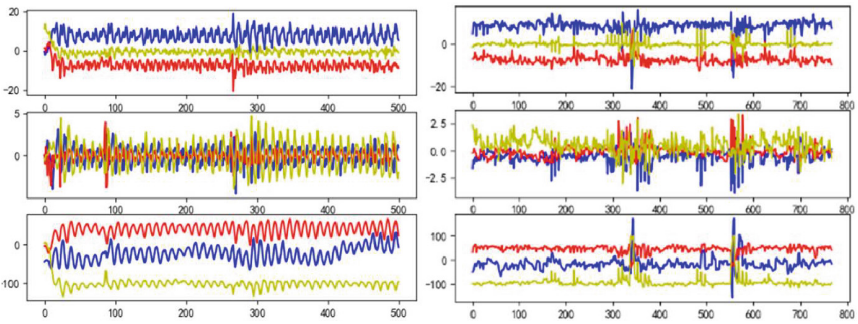

Fig. 5. Comparison between real data and generated data for walking action

The generated sensor data for different actions are shown in Fig. 6, with the left figure showing the model-generated data for sitting still and the right figure showing the model-generated data for walking actions. Visually, it can be seen that the model makes a good distinction between different types of actions, and the graphical features of the two are clearly different. When the person is sitting in a stationary state, the changes of the acceleration sensor, angular velocity sensor, and orientation sensor are very small, and the z-axis data of the acceleration sensor (yellow line in the figure) is close to the gravitational acceleration of 9.8 m/s2, which is in line with the objective law. When a person walks, the data of acceleration sensor, angular velocity sensor and direction sensor will change with the movement, and the generated data also conforms to this objective law.

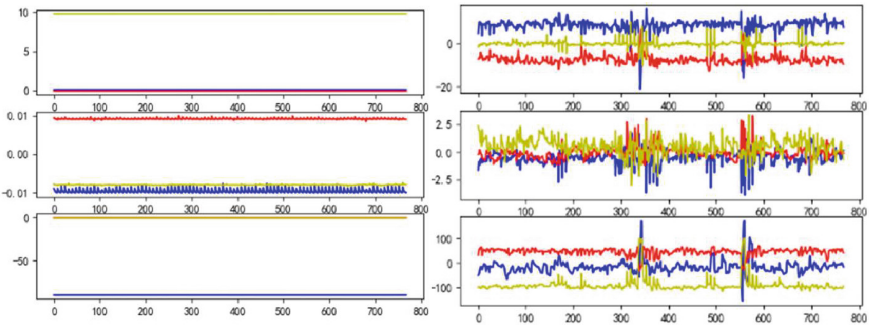

Fig. 6. Comparison of data generated by sitting still and walking

4.2 Classification Accuracy Assessment

The generation system proposed in this paper supports nine types of actions: going up stairs, going down stairs, standing, sitting, running, walking, jumping, playing games, and making phone calls. Due to the small number of sensors used in the current publicly available datasets in the HAR domain such as UCI HAR Dataset, HAPT Dataset, ADL Dataset, etc., they do not quite match with our needs. Therefore, the sensor dataset used in this experiment was collected by 50 students and teachers through our self-developed APP, which contains data from acceleration sensors, angular velocity sensors, and orientation sensors for the 9 types of actions mentioned above. In order to improve the classification accuracy of the generated data, this paper adds a threshold setting of 0.99 to the classification model proposed in Sect. 3. The classification accuracy (ACCURACY) of the generated data and real data is shown in Table 1.

Table 1. Classification accuracy of real data and generated data

Movements	True Data Accuracy	Generated Data Accuracy
Upstairs	94%	90%
Downstairs	99%	90%
Standing	100%	97%
Sitting	100%	100%
Running	100%	94%
Walking	99%	90%
Jumping	100%	90%
Playing games	100%	100%
Calling	93%	93%

It can be seen that the simulated data from the three sensors generated by the model achieves an accuracy of 90% and above under the behavioural action recognition task,

and this accuracy is also very close to the corresponding real data, thus confirming that the data generated by the model in this paper can simulate the characteristics of real data very well.

4.3 Personalised Evaluation

DTW (Dynamic Time Warping) is a method used to calculate the similarity between two time series. It finds the best match between two time series data and returns a distance value as a basis for similarity. Therefore, the DTW distance can be used to measure the personalised differences between two sequences. The larger the DTW distance, the more pronounced the personalised differences between the two actions.

In order to compare the diversity between the model-generated data and the diversity between the real data, we selected four non-stationary actions, namely running, walking, going up the stairs, and going down the stairs, and calculated the values of the real data between different users, and the DTW distance of the corresponding user-generated data. We then normalised the DTW distance by dividing it by the length of the data, and then divided it by the number of features of the sensor to get the final result, which is used to represent the diversity score of the action. The scoring results are shown in Table 2:

Table 2. Diversity score of generated data

Movements	Generate Data	Real Data
Upstairs	14.47	16.85
Downstairs	18.55	16.41
Running	26.14	7.29
Walking	9.08	6.06

It can be seen that the diversity scores of the generated data on the three actions of going up stairs, going down stairs, and walking are close to the real data, while the scores on the running action are much larger than the real data. This proves that the data we generated for different users are significantly different from each other, in line with real-life patterns.

4.4 Personalised Behavioural Data Generation

In total, we formed a user personalised sensor domain map containing 10 user roles, 31 labels and 58 keywords, and took the user role “undergraduate” as an example, and the results of the creation of its related nodes and relationships.

In order to work with the Markov chain, we need to use the Cypher query statement to query the corresponding personalised behavioural sensor keywords for the set user roles and labels, and then correspond to the status of the daily activities of different roles in the Markov chain. For example, by querying the personalised behaviour sensor keywords corresponding to the role “undergraduate student” and the tags “graduate

school" and "ordinary person", this paper generates the daily behaviour sensor data of 10 roles through the Markov chain. Through Markov chain, this paper generates daily behaviour sensor data for 10 roles, and sets several states for different roles. The specific states are shown in Table 3.

Table 3. 10 roles and corresponding states

Role	Status
Teacher	Sleeping, eating at home, eating,
Undergraduate	Going to work, leaving work, going to class, running in the playground, office/study
Graduate student	Sleeping, eating, going to class, picking up deliveries, running in the playground, exercising in the gym, studying in the library, playing on my mobile phone
Librarian	Sleeping, eating at home, eating,
Cleaner	Pick up delivery, gym exercise, playground running, library/research building study, play mobile phone
Delivery boy	Sleeping, eating at home, going to work, leaving work, sorting books, on duty, playing with mobile phone
Elderly people	100%
Athlete	Sleeping, eating, training (stretching), bathing/washing, resting (playing with mobile phone)
Bus driver	Sleeping, eating, driving, playing with mobile phone
Security guards	Sleeping, eating, going to/from work, patrolling, standing guard, playing with mobile phone

In the field of sensor data generation, the vast majority of research only focuses on the generation and recognition of single actions, without considering the generation of long-duration, comprehensive personalised data containing multiple actions according to different roles. In this experiment, based on the state transfer templates designed for 10 roles, we combine the generated single kind of action data of different roles into long-time sensor data that meets the characteristics of the roles by calling the generated single kind of action data.

Sample state transfer templates we generated are as follows:

{'6-7': {'eating': [('walking', 946), ('going upstairs', 23), ('sitting down', 1600), ('getting up', 1), ('walking', 45), ('going down', 33), ('walking', 945)]}

The template contains the time period in which the state occurs (6–7 am), the name of the state (eating), the action involved in realising the state and the corresponding time. We get the action and time by extracting the corresponding key-value pairs, and call the action data that has been generated for this role to stitch them together one by one to get the final sensor data for this user. The implementation is shown in Fig. 7:

In this way, we can generate personalised sensor data with different user behavioural characteristics and different lengths of time based on different state transfer templates.

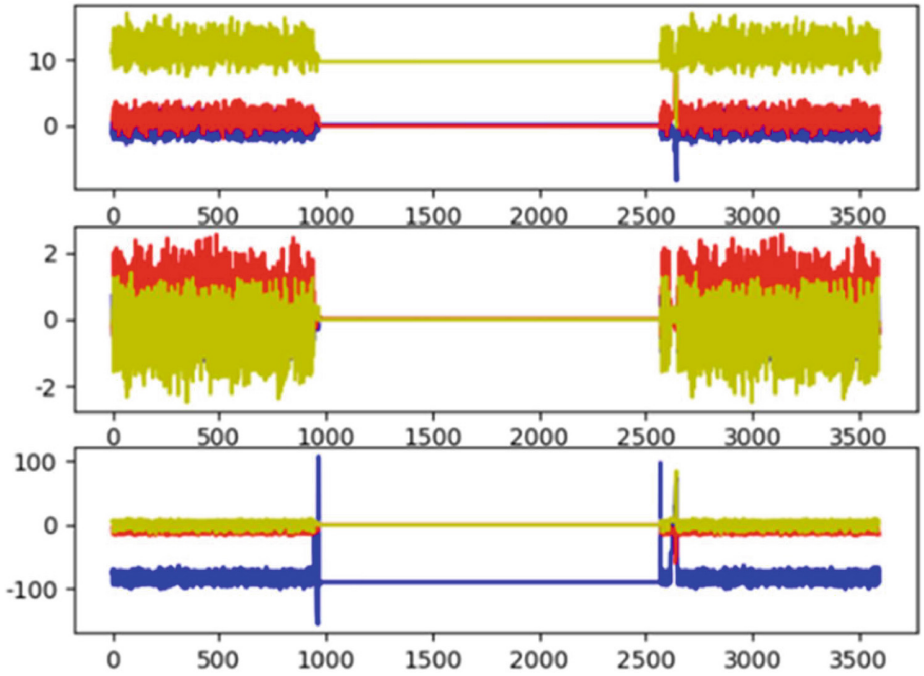

Fig. 7. User behavioural characteristics sensor data

Due to the large amount of data, the presentation with images is therefore not very aesthetically pleasing, and we will subsequently improve the visualisation method as well. In addition, the articulation between different actions is currently not very smooth, which is slightly different from the transition between different actions of real data, which is also something we will improve in the future.

5 Summary and Outlook

Through the above demonstration and illustration, our proposed method has significantly improved in terms of accuracy and diversity. Meanwhile, through the state transfer templates generated by matter-of-fact mapping, we have creatively generated a virtual identity personalised sensor dataset that is long-lasting, continuous, conforms to the logic of matter-of-fact mapping, and reflects the user's behavioural characteristics well. However, the sensor data generated by our proposed model still has some shortcomings, e.g., it appears to be more abrupt in the details of waveform images and the articulation transition between different action data is not smooth and natural. Our future research directions include improving the quality of data generation, generating longer samples with higher sampling frequency, and generating more personalised datasets based on more character templates. In addition, we will investigate privacy data generated by other sensors to increase the modality of the input data in our model.

Acknowledgements. This work is supported by the Major Projects of National Natural Science Foundation of China (Grant No.72293583), Research on Privacy Data Protection and Iatrogenesis Decision of IHS.

References

1. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. CoRR abs/1312.6114 (2013). n. pag
2. Goodfellow, I.J., et al.: Generative adversarial nets. In: NIPS (2014)
3. Esteban, C., et al.: Real-valued (Medical) time series generation with recurrent conditional GANs. arXiv abs/1706.02633 (2017). n. pag
4. Alzantot, M., Chakraborty, S., Srivastava, M.: SenseGEN: a deep learning architecture for synthetic sensor data generation. In: 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 188–193. IEEE (2017)
5. 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), Rio de Janeiro, Brazil, pp. 1–8 (2018). https://doi.org/10.1109/IJCNN.2018.8489106
6. Alzantot, M.F., et al.: PhysioGAN: training high fidelity generative model for physiological sensor readings. arXiv abs/2204.13597 (2022). n. pag
7. Roberts, A., et al.: A hierarchical latent vector model for learning long-term structure in music. In: International Conference on Machine Learning (2018)
8. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training GANs. In: Advances in Neural Information Processing Systems, pp. 2234–2242 (2016)
9. Chen, Y.Q., Wang, J.W., Gu, Y., Xiao, Y.L., Pan, H.N.: A sensor data generation method and system based on generative adversarial network. CN109086658B, Beijing, 8 June 2021
10. Chen, Y.-Q., Wang, J., Gu, Y., Wang, Y.-B., Zhang, Z.-P., Xiao, Y.-S.: A sensor data generation model and method based on conditional generative adversarial network. CN112884076A, Beijing, 1 June 2021
11. Lee, S.-S.: Derivation formula for Markov transfer probability matrix. Forecasting **1986**(01), 27–30+33 (1986)
12. Lee, Y.L., Lee, C.L.: Estimation and test of transfer probability. Math. Eng. **03**, 35–38 (1994)
13. Wen, S., Xu, M., Wang, F., Xu, Z., Sun, F.: A new method for estimating Markov state transfer probability matrices. Pract. Underst. Math. **44**(08), 164–169 (2014)
14. Nie, D., Chen, H., Mi, C., Peng, L.: Markov chain state probability transfer matrix correction algorithm. Stat. Decis. Making **03**, 14–17 (2013)
15. Lehmann, F.: Semantic networks. Comput. Math. Appl. **23**(25), 1–50 (l992)
16. Tian, Y.L., Li, X.: Evolutionary path analysis of online public opinion on the new coronary pneumonia outbreak based on matter-of-fact mapping. Intell. Theory Pract. **44**(3), 76–83 (2021)
17. Xu, H.: Study on the evolution path of multidimensional characteristics of online public opinion based on mapping. Intell. Sci. **40**(7), 48–54 (2022)