



Generation of Synthetic Drawing Samples to Diagnose Parkinson's Disease

Gennaro Gemito¹, Angelo Marcelli^{1,2}, and Antonio Parziale^{1,2}

¹ DIEM, University of Salerno, Via Giovanni Paolo II 132, 84084 Fisciano, SA, Italy
gemito.gen@gmail.com, {amarcelli, anparziale}@unisa.it

² AI3S Unit, CINI National Laboratory of Artificial Intelligence and Intelligent Systems, University of Salerno, Fisciano, SA, Italy

Abstract. The state-of-the-art artificial intelligence tools for automatic diagnosis of Parkinson's disease from handwriting require a lot of training samples from both healthy subjects and patients to exhibit impressive performance. Publicly available datasets include very few samples drawn by a small number of individuals and that limits the use of deep learning architectures. In this paper, we evaluate if the performance of a Convolutional Neural Network that recognizes the handwriting of Parkinson's disease patients can be improved by adding synthetic samples to the training set. In the experimentation, we synthetically generated dynamic signals of spirals and meanders through the use of a Recurrent Neural Network. The performance of the system was evaluated on the NewHandPD dataset and the results showed that the use of synthetic samples increases the recognition accuracy of the convolutional neural network.

Keywords: Data augmentation · Handwriting synthesis · Parkinson's disease · Handwriting analysis · CNN · RNN

1 Introduction

Parkinson's disease (PD) is a neurodegenerative disorder that affects dopaminergic neurons in the Basal Ganglia, whose death causes several motor and cognitive symptoms. PD patients show impaired ability in controlling movements and disruption in the execution of everyday skills, due to postural instability, the onset of tremors, stiffness and bradykinesia [10, 14, 23, 24].

There is no cure for the disease and the decline can only be somehow managed during its progression. This creates a critical need for improving the procedures and the tools for diagnosing them as early as possible.

The analysis of handwritten production has brought many insights for uncovering the processes occurring during both physiological and pathological conditions [3, 28, 29] and providing a non-invasive method for evaluating the stage of the disease [22].

The main advantage of diagnostic systems based on handwriting analysis with respect to other diagnostic procedures is that collecting handwriting samples is cheap and tests are easy to administer. Therefore, different artificial intelligence based approaches for the automatic identification of Parkinson's and Alzheimer's disease motor symptoms have been proposed, together with a variety of motor tasks administered to healthy individuals and patients [16, 20].

The desire of applying top-performing and most recent deep learning techniques to this application domain has highlighted the lack of a huge collection of handwriting samples. All the publicly available datasets include samples drawn by a reduced number of subjects and are, in some cases, unbalanced. Collecting data from patients is, in general, more complex than collecting data from healthy subjects mainly for two reasons: the need of reaching patients at their living places and the participation of specialized medical personnel during the administration of the diagnostic test.

To overcome the difficulty in collecting data from patients, in a very recent paper we have proposed to adopt machine learning tools based on *one-class* classification algorithms, i.e. algorithms capable of solving a two-class classification problem by learning the distinctive characteristics of only one class [17]. We have shown that is possible to reach state-of-the-art performance by training the Negative Selection Algorithm only with samples drawn by healthy subjects.

To overcome the more general problem of the scarcity of data and, therefore, to improve the performance of classifiers and to avoid overfitting of models, some papers investigated the usefulness of data augmentation and transfer learning to improve the classification performance [11, 15, 25]. The alterations introduced by data augmentation methods generate new samples that may not correspond to credible real samples. So, in this paper, we propose to increase the size of datasets by exploiting algorithms that are able to generate synthetic handwriting samples. To the best of our knowledge, that is the first time that these algorithms are applied to the application domain of diagnosis of neurodegenerative diseases. In particular, we evaluated the performance of the generator of synthetic samples based on a Recurrent Neural Network proposed by Alex Graves in [7].

The paper is organised as it follows. In Sect. 2, we review the previous works on synthetic handwriting generation and data augmentation and we show that there are different algorithms to generate synthetic handwriting but, up to now, they have not yet used to diagnose PD from handwriting. In Sect. 3, we briefly introduce the real data used in the experimentation, while in Sect. 4 we present the approach adopted to generate synthetic drawing samples. Section 5 introduces the CNN used as classifier and the approach adopted to convert signals to 2D images. Section 6 discusses the system that combines the synthetic generator of drawing samples and the classifier in order to select the best synthetic samples that will be used to train the final system. Sections 7 and 8 discuss the setup adopted to perform the experiments and the results we obtained. Eventually, Sect. 9 briefly discusses the results and Sect. 10 concludes the work.

2 Related Works

Many algorithms have been proposed in the literature for the generation of synthetic handwriting and they can be grouped in two different families: template-based approaches [4, 5, 9] and learning-based approaches [7, 12, 13]. The main difference between these two methodologies is that template-based approaches generate synthetic samples by perturbing real samples while learning-based approaches train neural networks to build a high-dimensional interpolation between training examples that will be used to generate synthetic samples.

Synthesis of handwriting samples has been applied to improve the performance of automatic systems for writer and signature verification [5, 21] and for handwriting recognition [1, 9, 12].

In the field of handwriting analysis for the diagnosis of neurodegenerative disorders, as Parkinson’s and Alzheimer’s disease, classical approaches of data augmentation have been recently implemented to boost the performance of deep learning networks. Classical approaches of data augmentation are those that have been applied in the more general field of computer vision and they include geometrical distortions and noise addition to real samples.

Rotations, flipping and contours were adopted in [15] to increase by a factor of 13 the cardinality of the dataset that was used to train the convolutional neural network AlexNet. In [11] the authors observed that rotation and thresholding had a negative impact on performance while illumination showed significantly better performance in comparison.

Data augmentation methods as jittering, scaling, time-warping, and averaging were applied directly to time series to increase the cardinality of the dataset used to train a CNN-BLSTM network [25]. The results showed that the combination of CNN-BLSTM and data augmentation was not effective when a single task was considered, but it increased the accuracy of 7.15% when the classification outputs per each task were combined.

3 Dataset of Real Sample

In this study, we used the NewHandPD dataset [19], a public dataset that includes handwritten data produced by 31 PD patients and 35 healthy subjects. The healthy group includes 18 male and 17 female individuals with ages ranging from 14 to 79 years old (average age of 44.05 ± 14.88 years), while the patient group includes 21 male and 10 female individuals with ages ranging from 38 to 78 years old (average age of 57.83 ± 7.85 years).

Each individual drew 12 samples: 4 spirals, 4 meanders, 2 circled movements, and 2 diadochokinesis. The dynamics of each sample were recorded by means of a Biometric Smart Pen (BiSP), while an image of the sample was available only for spirals, meanders and circles. A BiSP records 6 signals from as many sensors: voice signal $m(t)$, finger grip $gr(t)$, axial pressure of ink refill $p(t)$, tilt and acceleration in X direction $a_x(t)$, tilt and acceleration in Y direction $a_y(t)$, tilt and acceleration in Z direction $a_z(t)$.

In this study only the signals acquired with BiSP were taken into account while the images were not used. Moreover, we used only signals related to spirals and meanders, which are the classical motor tasks adopted to diagnose Parkinson's disease [30].

4 Generation of Synthetic Samples

4.1 The Method

The system we realized to generate synthetic drawing samples is based on the approach proposed by Alex Graves [7], which exploits Long Short-term Memory recurrent neural networks to generate complex sequences by predicting one data point at a time. The main idea behind the approach proposed by Graves is that RNN can be trained on real data sequences one step at a time so that the network can predict which point follows another one. Predicting the trajectory one point at a time has the advantage of increasing the diversity among the samples generated by the network.

It's worth noting that the network isn't trained to predict the next location of the pen, but to generate a probability distribution of what happens at the next instant of time, including whether the pen gets lifted up. In particular, a Gaussian mixture distribution is generated to predict the pen offset from the previous location while a Bernoulli distribution predicts if the pen stays on the paper or not.

This prediction is realized through the use of a *mixture density network* [2], which uses the outputs of a neural network to parameterise a mixture distribution. The combination of mixture density and RNN has the effect that the output distribution is conditioned not only on the current input, but on the history of previous inputs too.

In the original implementation proposed by Graves, each input vector x_t to the network is made-up of a real-valued pair $\{x_t^1, x_t^2\}$, which defines the pen offset from the previous input, and a binary x_t^3 that has value 1 if the vector ends a stroke. Each output vector y_t consists of the end-of-stroke probability e_t , a two-dimensional vector of means μ^j , a two-dimensional vector of standard deviation σ^j , correlations ρ^j and mixture weights π^j , which are scalars, for the M mixture components. The outputs of the network are transformed so that they satisfy the bounds related to the quantity they represent (for example, the real value used as correlation is bound between -1 and 1 with a hyperbolic tangent). Overall, the total number of outputs is equal to $(1 + M * 6)$.

The probability density $Pr(x_{t+1}|y_t)$ of the next input x_{t+1} given the output vector y_t is defined as follows:

$$Pr(x_{t+1}|y_t) = \sum_{j=1}^M \pi_t^j \mathcal{N}(x_{t+1} | \mu_t^j, \sigma_t^j, \rho_t^j) \begin{cases} e_t & \text{if } x_{t+1}^3 = 1 \\ 1 - e_t & \text{otherwise} \end{cases} \quad (1)$$

The probability given by the network to the input sequence x is

$$Pr(x) = \prod_{t=1}^T Pr(x_{t+1}|y_t) \quad (2)$$

and the sequence loss $L(x)$ used to train the network is the negative logarithm of $Pr(x)$

$$L(x) = - \sum_{t=1}^T \log(Pr(x_{t+1}|y_t)) \quad (3)$$

Once trained the RNN network is able to generate synthetic time sequences of a given length. As initial input, a vector of zeros is passed to the network and the end-of-stroke signal is turned on, signalling to the network that the next point it produces will be the start of a new stroke, rather than a continuation of an existing one. A zero state is also passed into the network for the initialization. After that, the network outputs the parameters of the probability distributions from which we randomly sample a set of values that represent a point of the synthetic sample. Afterwards, we repeat the loop and feed in the sampled point and network state back in as inputs, to get another probability distribution to sample from for the next point, and we repeat until we get the desired number of points.

4.2 The Implementation

The method described before was proposed to generate synthetic handwritten text, which consists of sequences of characters and then of words, and therefore, in the original implementation, the term end-of-stroke referred to the end of a pen-down.

Instead, in this paper, we want to synthesize drawing samples that are usually executed in a single pen-down. We know by handwriting generation theory that a pen-down is a superimposition of elementary movements, so, in our implementation, the end of a stroke corresponds to the end of an elementary movement.

The real samples were segmented in elementary movements by looking at the zero crossings of the tangential velocity along the y-axis [26]. Because the samples in NewHandPD include only the acceleration along the three axes, we needed to integrate these signals to detect the start and the end of elementary movements. Accelerations were filtered before integration, as suggested in another paper that used a smartpen similar to the one used for collecting NewHandPD data [27]. We filtered the acceleration signals between 0.5 Hz and 12 Hz with a 4th order Butterworth filter to remove the DC component related to slow oscillations and gravity and the frequencies beyond the range of relevant tremor components.

Given the time series available for each real sample, the inputs of the network were the acceleration along x and y instead of x and y as in Graves’ paper.

As with regards to the architecture of the network, we used a 2-layer stacked basic-LSTM network (no peephole connections) with 256 nodes in each layer. The number M of mixture components was set equal to 20, as in the original

Graves' paper. Overall, the network outputs 121 real values used to infer the probability distribution.

In our experiments, we adopted the implementation made available by David Ha at [8]. This implementation adopts mini-batches to fast the learning process. Mini-batches require to be the same length to be efficient and therefore time series were divided in sequences of length SL . Afterward, continuous portions of SL points were randomly sampled from the training sample and included in the mini-batches. Eventually, a dropout layer was included for each of the output layers of the LSTM to regularise training and reduce the overfitting.

5 Classification Stage

5.1 From Signal to Image

Let K be the total number of time series available for each handwriting sample and k the time series selected for the generation of the 2D image. Let n be the number of points of each time series.

The selected k time series of a handwriting sample are rearranged into a $(\sqrt{n \times k}, \sqrt{n \times k})$ square matrix Im that, in turn, is resized into a 64×64 image using the Lanczos resampling method. When $n \times k$ is not a square number, time series are truncated just enough to guarantee that Im has two natural numbers as dimensions.

Im is built by concatenating n arrays of k elements. In particular, the i -th array is made up of the value at time t_i of each time series. The arrays are horizontally concatenated so that the rows of Im are filled one by one. It is worth noting that the values of the time series are scaled in the range $[0, 255]$ so that each value can be represented by a grayscale pixel.

This approach for the generation of a 2D image from the time series of a sample is similar to the ones proposed in [19,25] but differs in the way Im is filled. In particular, the approaches in the literature concatenate the arrays so that the columns of the matrix, instead of rows, are filled one by one. Figure 1 shows an example of an image generated by following the approach described in this subsection.

5.2 Convolutional Neural Network

The 2D images representing the time series of handwriting samples are classified by a Convolutional Neural Network (CNN) whose architecture is taken from the neural network CIFAR-10 presented in [18]. The network has been fully implemented from scratch using the Python *TensorFlow2.0* library. Figure 2 shows a top-level view of the entire architecture and the hyper-parameters related to each layer.

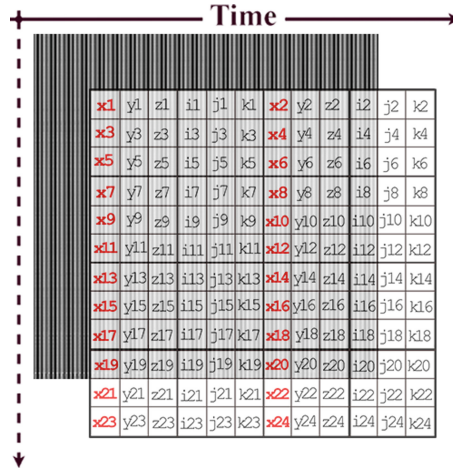


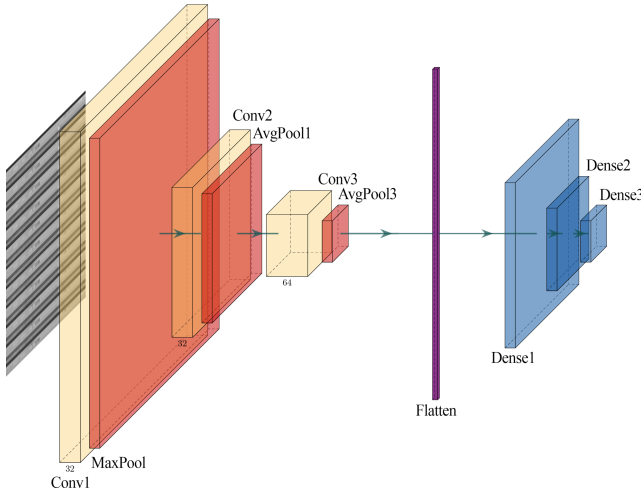
Fig. 1. An example of a grayscale 2D image obtained by $k = K = 6$ time series

6 Validation of Synthetic Data

The solution adopted in this work to assess the output of the RNN draws inspiration from the operation of Generative Adversarial Networks, or GANs [6]. GANs learn how to generate new data by adopting an architecture with two agents: a *Generator Model*, which generates a new example, and a *Discriminator Model*, which decides if the generated example is real or synthetic. Both the *Generator* and the *Discriminator* are Neural Networks: the Generator output is connected directly to the Discriminator input so, through backpropagation, the Discriminator's classification provides signals that the Generator uses to update its weights.

The main difference between standard GAN models and the approach proposed here is that the Generator doesn't learn from Discriminator's feedback since the two models are independent of each other. In fact, while in the GANs the two agents are connected through backpropagation so that the output of the Discriminator updates the weights of the Generator, in this work *Generator* and *Discriminator* are two separate subsystems: the RNN described in Sect. 4 generates synthetic data and, in a separate moment, these data are validated by the classifier described in Sect. 5 and trained on real data. So, the synthetic samples correctly classified by the CNN will be used to increase the dimension of the training set while the others will be discarded. Figure 3 shows the system implemented to validate synthetic samples.

It is worth noting that in our system the Discriminator consists of 5 CNNs and it classifies samples by a majority vote algorithm. The dataset of real samples is shuffled 5 times and each time a CNN is trained with 35% of the data.



- **Input Layer;**
- **Convolution Layer (1):**
 - Features: 32
 - Kernel size: 5×5
 - Stride: 1×1
 - Activation Function: ReLU
- **MaxPooling Layer:**
 - Pool size: 3×3
 - Stride: 2×2
- **Convolution Layer (2):**
 - Features: 32
 - Kernel size: 5×5
 - Stride: 1×1
 - Activation Function: ReLU
- **AveragePooling Layer (1):**
 - Pool size: 3×3
 - Stride: 2×2
- **Convolution Layer (3):**
 - Features: 64
 - Kernel size: 5×5
 - Stride: 1×1
 - Activation Function: ReLU
- **AveragePooling Layer (3):**
 - Pool size: 3×3
 - Stride: 2×2
- **Flatten;**
- **Dense Layer (1):**
 - Features: 64
 - Activation Function: Sigmoid
- **Dense Layer (2):**
 - Features: 10
 - Activation Function: Sigmoid
- **Dense Layer (3) - Output:**
 - Features: 1
 - Activation Function: Sigmoid

Fig. 2. Top-level view of the CNN used as classifier and hyper-parameters of each layer. The image was realized with Plotneuralnet.

7 Experimental Setup

7.1 Time Series Selection

We performed a preliminary experiment to verify if representing each sample with two time series instead of six could affect the performance of the classifier. In particular, we compared the performance of the network when the following two sets of time series were considered:

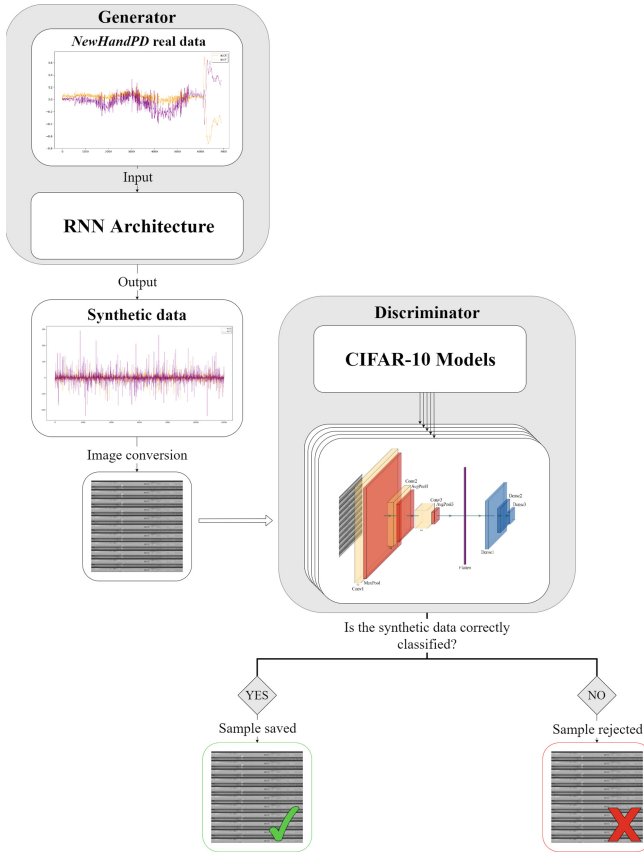


Fig. 3. The approach adopted to validate synthetic samples

- $\{m(t_i), gr(t_i), p(t_i), a_x(t_i), a_y(t_i), a_z(t_i)\}$;
- $\{a_x(t_i), a_y(t_i)\}$

Table 1 resumes all the choices we did to configure the experiment and the learning process of the convolutional neural network. Those values were selected after a fine-tuning process aimed at avoiding overfitting and maximizing the performance of the network on the validation set. The partition of the data in training, validation and test sets was made by guaranteeing that all the samples drawn by an individual were included in only one of the sets. We trained two different CNNs, one devoted to discriminating between PD and healthy subjects by looking at spirals and the other by looking at meanders.

Table 2 shows the results obtained by the two CNNs when images are generated by one of the two sets of time series. For each drawing task, we evaluated if the difference between the classification accuracies was statistically significant or not by performing a Wilcoxon test, with a level of significance $\alpha = 0.05$. As shown in the table, differences are not statistically significant, both for spirals

and meanders, but the p-value obtained in the case of spirals is close to the level of significance and the evidence for the null hypothesis that using two time series instead of six has no effect on the accuracy is weaker.

We chose to represent samples only with two time series, tilt and acceleration in X direction $a_x(t)$ and tilt and acceleration in Y direction $a_y(t)$.

Table 1. Experimental setup to classify 2D images with the CNN

Parameter	Value
Kernel initializer	Glorot normal
Bias initializer	0
Pseudorandom number generators	Fixed seeds
Training/validation/test size	35%/15%/50%
k-fold cross validation	5-fold
Batch size	5
Optimization algorithm	SGD
Learning rate	2×10^{-5}
Momentum	0.9
Nesterov momentum	True
Loss	Binary cross entropy
Early stopping	Min validation loss
Epochs	10000

Table 2. Performance of the CNN when two different sets of time series are considered. The last column reports the p-value of the Wilcoxon statistical test. 2 time series: $\{a_x(t_i), a_y(t_i)\}$, 6 time series: $\{m(t_i), gr(t_i), p(t_i), a_x(t_i), a_y(t_i), a_z(t_i)\}$.

Task	Accuracy (%)		p-value
	2 time series	6 time series	
Spirals	70.75 ± 3.22	77.01 ± 3.25	0.07961
Meanders	74.42 ± 7.81	74.72 ± 2.62	0.89274

7.2 Generation and Validation of Synthetic Samples

We trained 4 different RNNs, which were implemented according to the architecture described in Sect. 4: the first synthesized spiral drawn by healthy subjects, the second synthesized spiral drawn by PD patients, the third and the fourth synthesized meanders drawn by healthy and PD patients, respectively.

Table 3 reports the hyper-parameter values used for these networks, which were fine-tuned with the aim of minimizing the loss on the validation set.

Once the 4 networks were trained, each of them generated SS valid samples. The validity of a sample was verified with the system that combines the synthetic generator and the classifier, as described in Sect. 6. We evaluated seven conditions with $SS = [100, 300, 500, 600, 800, 1000, 1500]$.

The length of each sample (the number of its points) generated by one of the RNNs was chosen so that the distribution of synthetic samples per number of points was similar to the distribution of real samples per number of points.

Table 3. RNN hyper-parameters to generate synthetic samples

Parameter	Value
RNN hidden state	256
Number of stacked cells (layers)	2
Cell type	LSTM
Sequence length SL	512
Number of epochs	300
Learning rate	0.005
Number of mixture M	20
Dropout keep probability	0.8
Training/validation set	70%/30% of total samples
Loss function	Log likelihood loss

7.3 Data Augmentation and Classification

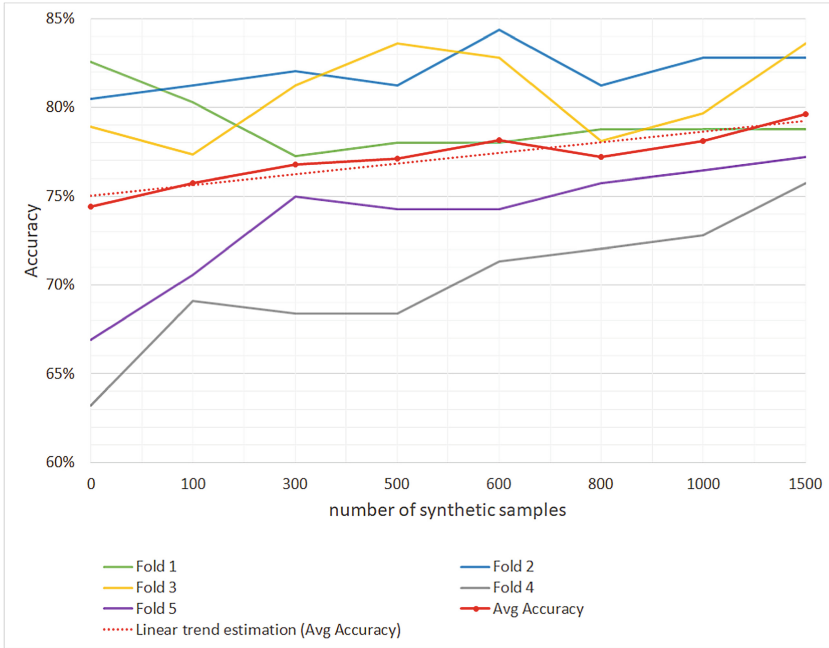
Two CNNs were trained with both real and synthetic samples to diagnose Parkinson’s disease: one processed spirals, the other one meanders. The two CNNs were trained using the hyper-parameters reported in Table 1, the same ones we used to train the network with only real samples.

The $2 * SS$ synthetic spirals (SS per class) and the $2 * SS$ synthetic meanders were added to the 50% of real samples used in the training phase, which were split between training and validation sets. Synthetic samples were not included in the test set.

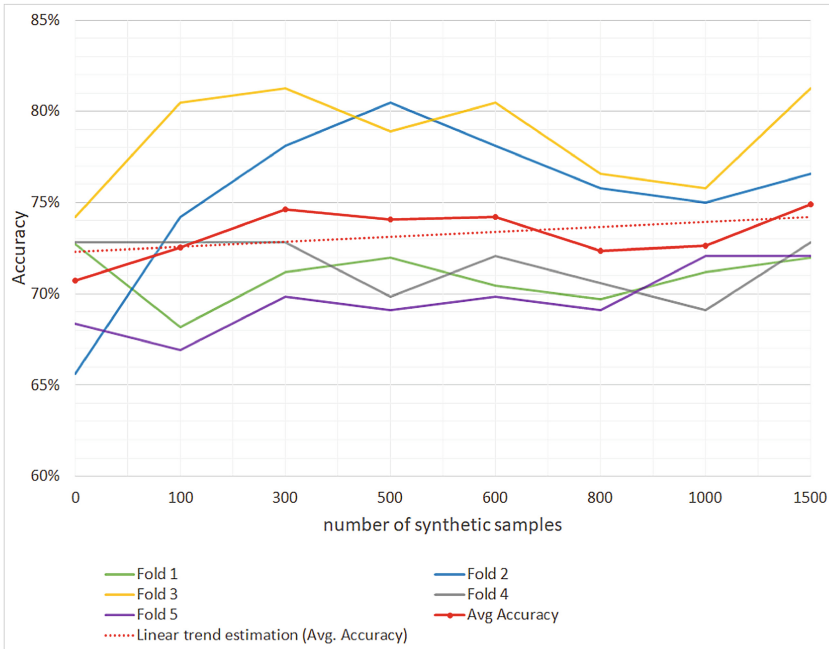
The performance was measured by averaging on 5 training of the CNNs. Every time, the real dataset was shuffled and 50% of subjects were kept apart as test set. So, the samples drawn by an individual were included in the training or in the test set but never in both of them.

8 Results

Figure 4 shows how the accuracy of the system varies as the number of synthetic samples varies from 0 to 1500, both for spirals and meanders. Figure 4a shows



(a) meanders



(b) spirals

Fig. 4. Accuracy vs number of synthetic samples for meanders and spirals

Table 4. Accuracy (Acc.), Sensitivity (Sens.) and Specificity (Spec.) of the CNN when it is trained without or with synthetic samples.

	No synthetic samples			1500 synthetic samples		
	Acc. (%)	Sens. (%)	Spec. (%)	Acc. (%)	Sens. (%)	Spec. (%)
Spirals	70.75 ± 3.22	68.69 ± 5.19	72.62 ± 3.17	74.93 ± 3.58	74.08 ± 5.54	76.09 ± 3.26
Meanders	74.42 ± 7.81	75.60 ± 6.22	73.54 ± 8.10	79.63 ± 3.09	78.77 ± 4.28	80.32 ± 6.78

an evident increasing linear trend of the average accuracy as the number of synthetic meanders in the training set increases. Moreover, the increase in the number of synthetic meanders reduces the differences in the performance of the system when it is trained with a different set of data. Figure 4b shows that these comments are still valid for spirals even though less evident.

Table 4 reports the results obtained when the CNN described in Sect. 5 was trained only with real samples or with the addition of 1500 synthetic samples (15% of which were used as validation).

The use of synthetic spirals had the effect of increasing accuracy, sensitivity and specificity of 4.18%, 5.39% and 3.47%, respectively. The use of synthetic meanders had the effect of increasing accuracy, sensitivity and specificity of 5.21%, 3.17% and 6.78%, respectively. Moreover, the standard deviation of the system trained with real and synthetic samples is lower than the variability of the system trained with real data only. In particular, the standard deviation of the accuracy is more than halved when synthetic samples are used.

9 Discussion

The RNN used to generate synthetic samples does not directly output the next location of the pentip but the parameters of mixture distributions that are sampled to predict the next point of the trajectory. Sampling one point at a time from probability distributions favours the diversity between real and synthetic samples. A preliminary analysis confirmed that the synthetic samples differ from the ones in the test set although the entire real dataset was used to train and validate the RNN. Nevertheless, this aspect will be further investigated in future works by keeping the test set apart from the data used to train the generator network.

The difference in performance when the system is trained with meanders instead of spirals could be ascribed to the selection of time series. In fact, as we can see in Table 2, the use of two time series instead of six has no effect on meanders but lower the accuracy when spirals are evaluated, even though it is not statistically significant. This aspect needs to be investigated in the future by increasing the number of experiments.

10 Conclusions

Publicly available datasets are made-up of samples handwritten by small numbers of subjects. This aspect limits the use of top-performing deep learning algorithms that need a huge amount of data to correctly classify samples. To solve this problem, we have proposed to use algorithms for handwriting synthesis to increase the size of the training set.

Preliminary results have confirmed that the addition of synthetic samples to the training set increases the performance of a basic convolution neural network and reduce the variability of performance as the training set varies.

In our future investigations, we will aim at performing a systematic investigation that will take into account different neural network architectures as well as samples corresponding to other motor tasks, as for example words and sentences. Moreover, we plan to combine template-based and learning-based methods to increase the diversity of synthetic samples. Eventually, we will exploit a correlation analysis between basic handwriting features and the clinical state of participants in order to evaluate if it is possible to generate synthetic samples that denote different stages of PD, from early to severe stages.

References

1. Bhattacharya, U., Plamondon, R., Dutta Chowdhury, S., Goyal, P., Parui, S.K.: A sigma-lognormal model-based approach to generating large synthetic online handwriting sample databases. *Int. J. Doc. Anal. Recogn. (IJDAR)* **20**(3), 155–171 (2017). <https://doi.org/10.1007/s10032-017-0287-5>
2. Bishop, C.: Mixture density networks. Technical report NCRG/94/004, Aston University, January 1994. <https://www.microsoft.com/en-us/research/publication/mixture-density-networks/>
3. Broderick, M.P., Van Gemmert, A.W., Shill, H.A., Stelmach, G.E.: Hypometria and bradykinesia during drawing movements in individuals with Parkinson's disease. *Exp. Brain Res.* **197**(3), 223–233 (2009). <https://doi.org/10.1007/s00221-009-1925-z>
4. Carmona-Duarte, C., Ferrer, M.A., Parziale, A., Marcelli, A.: Temporal evolution in synthetic handwriting. *Pattern Recogn.* **68**, 233–244 (2017)
5. Diaz, M., Ferrer, M.A., Eskander, G.S., Sabourin, R.: Generation of duplicated off-line signature images for verification systems. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(5), 951–964 (2016)
6. Goodfellow, I.J., et al.: Generative adversarial networks (2014)
7. Graves, A.: Generating sequences with recurrent neural networks (2013). <https://doi.org/10.48550/ARXIV.1308.0850>. <https://arxiv.org/abs/1308.0850>
8. Ha, D.: Write RNN TensorFlow (2018). <https://github.com/hardmaru/write-rnn-tensorflow>
9. Haines, T.S., Mac Aodha, O., Brostow, G.J.: My text in your handwriting. *ACM Trans. Graph. (TOG)* **35**(3), 1–18 (2016)
10. Jankovic, J.: Parkinson's disease: clinical features and diagnosis. *J. Neurol. Neurosurg. Psychiatry* **79**(4), 368–376 (2008)

11. Kamran, I., Naz, S., Razzak, I., Imran, M.: Handwriting dynamics assessment using deep neural network for early identification of Parkinson's disease. *Future Gener. Comput. Syst.* **117**, 234–244 (2021)
12. Kang, L., Riba, P., Wang, Y., Rusiñol, M., Fornés, A., Villegas, M.: GANwriting: content-conditioned generation of styled handwritten word images. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) *ECCV 2020*. LNCS, vol. 12368, pp. 273–289. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58592-1_17
13. Kumar, K.M., Kandala, H., Reddy, N.S.: Synthesizing and imitating handwriting using deep recurrent neural networks and mixture density networks. In: 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1–6. IEEE (2018)
14. Marsden, C.: Slowness of movement in Parkinson's disease. *Mov. Disord. Off. J. Mov. Disord. Soc.* **4**(S1), S26–S37 (1989)
15. Naseer, A., Rani, M., Naz, S., Razzak, M.I., Imran, M., Xu, G.: Refining Parkinson's neurological disorder identification through deep transfer learning. *Neural Comput. Appl.* **32**(3), 839–854 (2020). <https://doi.org/10.1007/s00521-019-04069-0>
16. Parziale, A., Senatore, R., Della Cioppa, A., Marcelli, A.: Cartesian genetic programming for diagnosis of Parkinson disease through handwriting analysis: performance vs. interpretability issues. *Artif. Intell. Med.* **111**, 101984 (2021)
17. Parziale, A., Della Cioppa, A., Marcelli, A.: Investigating one-class classifiers to diagnose Alzheimer's disease from handwriting. In: Sclaroff, S., Distanto, C., Leo, M., Farinella, G.M., Tombari, F. (eds.) *ICIAP 2022*. LNCS, vol. 13231, pp. 111–123. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-06427-2_10
18. Pereira, C.R., et al.: A new computer vision-based approach to aid the diagnosis of Parkinson's disease. *Comput. Methods Programs Biomed.* **136**, 79–88 (2016)
19. Pereira, C.R., Weber, S.A.T., Hook, C., Rosa, G.H., Papa, J.P.: Deep learning-aided Parkinson's disease diagnosis from handwritten dynamics. In: Proceedings of the Conference on Graphics, Patterns and Images, pp. 340–346. IEEE (2016)
20. Pereira, C.R., Pereira, D.R., Weber, S.A., Hook, C., de Albuquerque, V.H.C., Papa, J.P.: A survey on computer-assisted Parkinson's disease diagnosis. *Artif. Intell. Med.* **95**, 48–63 (2019)
21. Pignelli, F., Costa, Y.M.G., Oliveira, L.S., Bertolini, D.: Data augmentation for writer identification using a cognitive inspired model. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) *ICDAR 2021*. LNCS, vol. 12824, pp. 251–266. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86337-1_17
22. Senatore, R., Marcelli, A.: A paradigm for emulating the early learning stage of handwriting: performance comparison between healthy controls and Parkinson's disease patients in drawing loop shapes. *Hum. Mov. Sci.* **65**, 89–101 (2019)
23. Sheridan, M., Flowers, K., Hurrell, J.: Programming and execution of movement in Parkinson's disease. *Brain* **110**(5), 1247–1271 (1987)
24. Stelmach, G.E., Teasdale, N., Phillips, J., Worringham, C.J.: Force production characteristics in Parkinson's disease. *Exp. Brain Res.* **76**(1), 165–172 (1989). <https://doi.org/10.1007/BF00253633>
25. Taleb, C., Likforman-Sulem, L., Mokbel, C., Khachab, M.: Detection of Parkinson's disease from handwriting using deep learning: a comparative study. *Evol. Intell.* (2020). <https://doi.org/10.1007/s12065-020-00470-0>
26. Teulings, H.L.: Handwriting movement control. In: Heuer, H., Keele, S.W. (eds.) *Motor Skills, Handbook of Perception and Action*, vol. 2, pp. 561–613. Academic Press (1996). [https://doi.org/10.1016/S1874-5822\(06\)80013-7](https://doi.org/10.1016/S1874-5822(06)80013-7)

27. Toffoli, S., et al.: A smart ink pen for spiral drawing analysis in patients with Parkinson's disease. In: 2021 43rd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC), pp. 6475–6478 (2021)
28. Tucha, O., et al.: Kinematic analysis of dopaminergic effects on skilled handwriting movements in Parkinson's disease. *J. Neural Transm.* **113**(5), 609–623 (2006). <https://doi.org/10.1007/s00702-005-0346-9>
29. Van Gemmert, A., Adler, C., Stelmach, G.: Parkinson's disease patients undershoot target size in handwriting and similar tasks. *J. Neurol. Neurosurg. Psychiatry* **74**(11), 1502–1508 (2003)
30. Vessio, G.: Dynamic handwriting analysis for neurodegenerative disease assessment: a literary review. *Appl. Sci.* **9**(21), 4666 (2019)