Computer-Aided Music Composition with LSTM Neural Network and Chaotic Inspiration

Andrés E. Coca[1], Débora C. Corrêa[2] and Liang Zhao[1]

***Abstract*— In this paper a new neural network system for composition of melodies is proposed. The Long Short-Term Memory (LSTM) neural network is adopted as the neural network model. We include an independent melody as an additional input in order to provide an inspiration source to the network. This melody is given by a chaotic composition algorithm and works as an inspiration to the network enhancing the subjective measure of the composed melodies. As the chaotic system we use the Hénon map with two variables, which are mapped to pitch and rhythm. We adopt a measure to conduct the degree of melodiousness (Euler's *gradus suavitatis*) of the output melody, which is compared with a reference value. Varying a specific parameter of the chaotic system, we can control the complexity of the chaotic melody. The system runs until the degree of melodiousness falls within a predetermined range.**

I. Introduction

The reproduction and generation of music has attracted researches a long time ago. Music composition by computers, specially, date back to the 1950's with the use of Markov chains to generate melodies. Since then, several studies have been conducted in order to achieve a computational music composition system that could, as much as possible, catch the human mind, skills, and creativity [1].

For musical computation, artificial neural networks (ANNs), as well as other systems that involve machine learning, are able to learn patterns and features present in the melodies of the training set and to generalize them to compose new melodies. In this context, we can find many approaches that use ANNs in music learning and composition [2]. Among different ANNs, the recurrent neural networks (RNNs) are particularly considered for music computation, since they contemplate feedbacks units and delay operators, which allow the incorporation of nonlinearity and dynamical aspects to the model. This makes the RNNs specially interesting mechanisms in the analysis of temporal time series. RNNs are usually trained with the Back-Propagation-Through-Time algorithm (BPTT) that implements a gradient-descent based learning method. However, previous gradient methods share a problem: over time, as gradient information is passed backward to update weights whose values affect later outputs, the error/gradient information is continually decreased or increased by weight-update scalar values [3]. This means that the temporal evolution of the path integral over all error signals flowing back in time exponentially depends on the magnitude of the weights. For this reason, standard recurrent neural networks fail to learn long time lags between relevant input and target events [4].

To overcome this drawback, the Long-Short-Term Memory (LSTM) network were designed [5]. The LSTM network is an alternative architecture for recurrent neural network inspired on the human memory systems. The principal motivation is to solve the vanishing-gradient problem by enforcing constant error flow through "constant error carrousels" (CECs) within special units, permitting the non-decaying error flow back into time [5]. The CECs are units responsible for keeping the error signal. This enables the network to learn important data and store them without degradation of long period of time.

Studies on musical composition using LSTM found in the literature started in 2002 with the paper of Eck and Schmidhuber [6]. The authors used the LSTM to learn musical forms of blues. In a second stage, the network is conducted to compose new melodies on blues style without losing the relevant structures over long time. Two additional experiments are further realized: in the first one, the network learns a sequence of chords and, in the second, it learns to play the chords along with the melody [7]. In [8] a computer program named ALICE used the LSTM network to compose music based on selection of pieces.

Recently, the LSTM network was used for music composition in [9], where the authors compare the performance of MLPs trained with the BPTT algorithm and LSTM networks in the task of composing new melodies. They used a nature-based inspiration to complement the application stage, in order to enhance the variability of the composed melodies. However, this method has some disadvantages: it requires the creation and storage of images in a dataset; and it requires the preprocessing of each image, with is time consuming. A similar works is in [10], where the RNN is trained with the BPTT algorithm and as the inspiration is provided by a chaotic dynamical system.

In the context of music composition, it is usual and important to determine the quality of the final melody. In other words, it might be interesting to tune the characteristics of the composed melody with respect to some set of criteria that evaluate the melody as "good". However, previous works quantifying the subjective characteristics of the created melodies are not known beforehand. In this manner, the aim of this paper is the proposal of a LSTM neural network based system for computer-aided musical composition in which the quality of the composed melody is evaluated. The inspiration source comes from a chaotic system. Due to the

[1] Andrés E. Coca and Liang Zhao are with the Institute of Mathematics and Computer Science, University of São Paulo, São Carlos - SP, Brazil, email:{aecocas,zhao}@icmc.usp.br.

[2] Débora C. Corrêa is with the Institute of Physics, University of São Paulo, São Carlos - SP, Brazil, email:{debcris.cor}@gmail.com.

characteristics of chaotic systems, it is possible to get infinite variations through arbitrarily small changes in the parameters of the system. As consequence, we can get many melodies with different characteristics and complexity. For establishing the chaotic melody, we use the algorithm proposed by Coca et. al. [11], and the Hénon map that has two variables: the first variable is used for determining the pitch, and, the second, the rhythm. A specific parameter of the chaotic system is varied, making it possible to control the complexity of the chaotic melody. The musical composition system is able to compose new melodies with a degree of melodiouness predefined. This subjective property is obtained for each output melody and compared with a reference value. The difference between the output and the reference value is used to modify a parameter of the dynamical system. By doing so, we can define a strategy to control the pitch-related complexity of the inspiration melody. The system runs until the value of the characteristic falls within a range of the reference value. With this the characteristic of the output melody is controlled by the chaotic inspiration source.

The overall organization of the paper is as follows: Section II describes the LSTM network; Section III presents the algorithm of musical composition; Sections IV and V show the full compositional system and the melodic measures used; simulation results are shown in Section VI. Finally, Section VII presents the principal conclusions and possibilities of future works.

II. The traditional LSTM network

Traditional RNNs are usually characterized as networks in which the outputs of the hidden units at time t are feedback to the same units at time $t+1$ (Fig. 1(a)). However, as mentioned in Introduction, they can suffer from the vanishing-gradient problem [4]. The LSTM network was developed to minimize this drawback by offering the memory block: the basic unit of the hidden layer (Fig. 1(b)).

A memory block includes one or various memory cells, a pair of gating units and output units (according to the quantity of memory cells). Figure 2 details a memory block with one memory cell. We can also observe a self-connected linear unit called “Constant Error Carrousel” (CECs) in each memory cell. This unit is fundamental to pass the error signals through the networks, minimizing the vanishing error problem. More details about the relationship between the CECs and the error/gradient signals can be verified in [5].

A. *Forward Pass*

The forward pass of the LSTM network is quite similar to the equivalent step in traditional RNNs, except that now the hidden unit comprises sub-units in which the signal is propagated. Due to space limitation, we will briefly describe the forward pass. The reader is invited to consider reference [4] and [5] for more details about the LSTM forward and backward algorithm.

Consider again Figure 2. Basically, the cell state s_c is updated according to its current state and the three connections: net_c, that reflects the signals from the input layer; net^{in} and

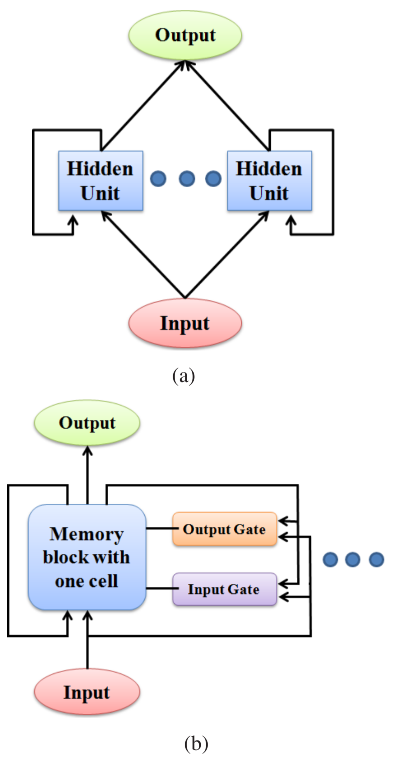

Fig. 1. ANNs architectures. (a) The RNN network with one recurrent hidden layer. (b) The LSTM network with memory blocks in the hidden layer (only one is shown).

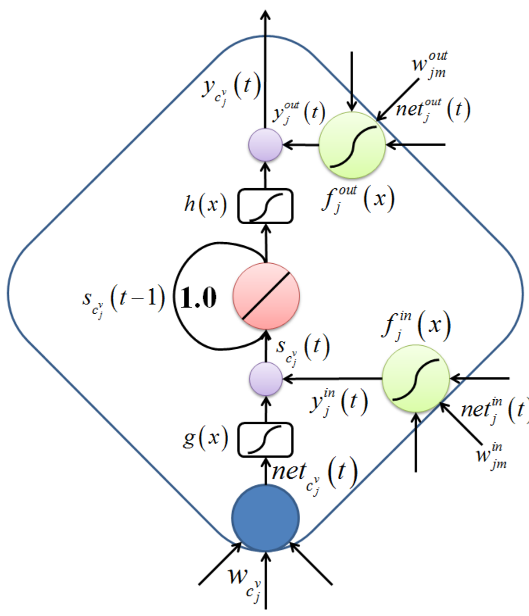

Fig. 2. The LSTM memory block with one memory cell.

net^{out} , that represent the connections from input and output gates respectively.

As usual, a time step t involves updating all units and computing the error signals in order to adjust the weights. The activation of the input gate $y_j^{in}(t)$ and the output gate $y_j^{out}(t)$ is generally defined by sigmoid function on a weighted sum of the inputs $net^{out}(t)$ and $net_j^{in}(t)$, both received via recurrent links from the memory blocks and from the external inputs to the network:

$$net_j^{in}(t) = \sum_m w_{jm}^{in} y_m(t-1) + b_j^{in} \tag{1}$$

and

$$y_j^{in}(t) = f_j^{in}(net_j^{in}(t)) \tag{2}$$

Accordingly:

$$net_j^{out}(t) = \sum_m w_{jm}^{out} y_m(t-1) + b_j^{out} \tag{3}$$

and

$$y_j^{out}(t) = f_j^{out}(net_j^{out}(t)) \tag{4}$$

where b_j^{in} and b_j^{out} are the unit weights for the gates.

The memory blocks are indexed by j; memory cells in block j by c_j^v; and w_{lm} is the weight of the connection from unit m to unit l.

The gates also use logistic sigmoid function $f(x)$ (generally with range [0,1]) as follows:

$$f(x) = \frac{1}{1+e^{-x}} \tag{5}$$

The inputs of c_j^v are multiplied by weights $w_{c_j^v m}$ (from input m to cell c_j^v) as follows:

$$net_{c_j^v}(t) = \sum_m w_{c_j^v m} y_m(t-1) \tag{6}$$

Following the sequence, a sigmoid function $g(x)$ ranging from -2 to 2 is applied in $net_{c_j^v}(t)$:

$$g(x) = \frac{4}{1+e^{-x}} - 2 \tag{7}$$

Consequently, the internal state of memory cell c_j^v is:

$$s_{c_j^v}(0) = 0; s_{c_j^v}(t) = s_{c_j^v}(t-1) + y_j^{in}(t) g(net_{c_j^v}(t)) \tag{8}$$

for $t > 0$. Finally, the cell output $y_{c_j^v}$ is:

$$y_{c_j^v} = y_j^{out}(t) h(s_{c_j^v}(t)) \tag{9}$$

where $h(x)$ is usually a sigmoid function ranging from $[-1, 1]$:

$$h(x) = \frac{2}{1+e^{-x}} - 1 \tag{10}$$

Now consider a network with a standard input layer, a hidden layer consisting of memory blocks, and a standard

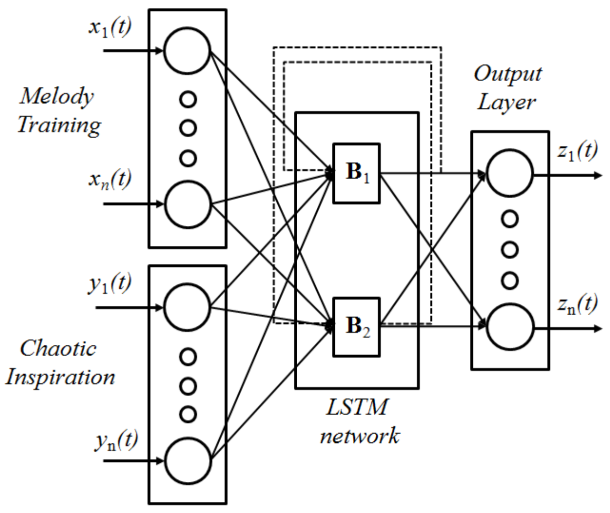

Fig. 3. The LSTM network witch chaotic inspiration.

output layer. The typical network's output $y_k(t)$ is the weighted sums of $net_k(t)$ that is propagated through a sigmoid function $f(x)$ (Eq.(5)) as follows:

$$net_k(t) = \sum_m w_{km} y_m(t-1) \tag{11}$$

$$y_k(t) = f_k(net_k(t)) \tag{12}$$

where m represents all units feeding the output units (generally, it comprises all the cells in the hidden units and the input units).

B. The LSTM network with chaotic inspiration

We can adapt the input layer of the LSTM network dividing the input units in two groups: the melodic units, and the chaotic units. The melodic units represent the melody supposed to be learned, and the chaotic units represent the inspiration given by the chaotic system.

Figure 3 shows the resulting LSTM recurrent network with two memory blocks. The melodic input units are represented by x_n, and y_n accounts for the inputs for the chaotic inspiration generated by the algorithm of composition described in Section III. The network outputs are represented by z_n. The cycles of thirds representation is used as the input data, established in the form of seven bits [12]. Each hidden unit represents a memory block with one memory cell as illustrated in Figure 2.

III. Chaotic Inspiration Algorithm

The algorithm developed in [11] uses the numerical solution of a nonlinear dynamical system. In this paper dynamical system with one and two variables are used. The first variable $x(t)$ is associated with the extraction of musical pitches and the second variable $y(t)$ with the rhythmic durations for the input inspiration. Data transformations of this variable are described in the following subsections.

A. Extraction of Frequencies and Musical Notes

The extraction of frequencies and musical notes is divided into three steps as follows:

1) Part 1: Generation of the Musical Scale: With the tuning factor λ (the inverse of the number of tone divisions, $\lambda = 1/\Delta$) and the number of octaves k, a vector s of dimension $(p+1)$ can be constructed. It contains the frequency ratios of an equal temperament scale and is generated by the following geometric sequence:

$$s_i = \ 2^{\lambda(i-1)/6}, \quad 0 < i \leq p+1 \tag{13}$$

where $p = 6k/\lambda$ is the total number of notes of a chromatic scale with tuning factor λ and k octaves. With the number of semitones (s), tones (t) and tone and a half (t_m) that form the architecture of the desired musical scale ξ, the components of the binary membership vector $\mathbf{v}$ of the desired scale are determined, in which an element 1 or 0 indicates, respectively, whether or not a note of desired scale is contained in the chromatic scale. The membership vector of a scale $\mathbf{v}$ acts as a filter on the vector $\mathbf{s}$, keeping only the intervals that belong to the scale. The result of this operation is the vector $e_i = \ v_i \cdot s_i, \quad 0 < i \leq (p+1)$. The elements in the vector $\mathbf{e}$ equal to zero are eliminated. Thus, we obtain a vector $\mathbf{g}$ with n elements.

2) Part 2: Variable Normalization: The variable $x(t)$ should be normalized with respect to the vector $\mathbf{g}$, because the range of the variable $x(t)$ is different from the range of the data vector $\mathbf{g}$. The normalization process, defined as $x_n = \gamma(x(t), \mathbf{g})$, consists of scaling and translating the numerical solution of variable $x(t)$ in order to adapt it to the data vector $\mathbf{g}$. In this way, we obtain the normalized x_n equal to $x_n(t) = \alpha x(t) + \beta$, where α is a scaling factor, calculated by the following formula:

$$\alpha = \frac{2^k - 1}{\max(x(t)) - \min(x(t))} \tag{14}$$

where k indicates the extent of the scale in octaves (note that $\max(\mathbf{g}) = 2^k$ and $\min(\mathbf{g}) = 1$). Similarly, the variable β is a translation factor and it is determined by the following equation:

$$\beta = -\alpha \min(x(t)) + \min(\mathbf{g}) = -\alpha \min(x(t)) + 1 \tag{15}$$

In this way, we get a variable in the range $1 \leq x_n(t) \leq 2^k$.

3) Part 3: Mapping to the Closest Value: Once the variable is normalized, we determine the closest value in the vector $\mathbf{g}$ for each value x_n, getting a match between the data vector x_n with the notes of the musical scale specified in ξ. Then a matrix $\mathbf{D}$ of dimension $c_{x_n} \times n$ is built with Eq. (16), such that c_{x_n} represents the number of elements of a piece of numerical solution of x_n and n is the number of musical notes and the indexes i and j are in the ranges $0 < i < c_{x_n}$ and $0 < j \leq n$, respectively.

$$D_{i,j} = \begin{cases} 0, & \textbf{if} \left| x_{n_j}(t) - g_i \right| \leq \mu \\ x_{n_j}(t), & \textbf{if} \left| x_{n_j}(t) - g_i \right| > \mu \end{cases} \tag{16}$$

where the threshold value μ, calculated by $\mu = 2^{\frac{\lambda}{6}} - 1$ (with a tuning factor $\lambda = 0.5$), is equal to the minimal interval generator. Then we generate a new vector $\mathbf{h}$ of size $c_{x_n} \times 1$ that holds the position of the minimum values for each row of the matrix $\mathbf{D}$. The knowledge of tonic is required for the conversion of the variable $x_n(t)$ to the musical space. The tonic frequency $\delta_{\tau,o}$ with the musical tone τ in the octave o can be obtained as follows:

$$\delta_{\tau,o} = 55 \cdot 2^{\frac{\tau+12o-10}{12}} \tag{17}$$

With the indexes of $\mathbf{h}$ and frequency of tonic $\delta_{\tau,o}$, we can calculate the frequencies of musical notes corresponding to the variable $x_n(t)$ equal to $f_i = \delta_{\tau,o} \cdot g_{h_i}, 0 < i \leq c_{x_n}$. In order to view the score of the melody, these frequencies must be converted to the values of musical notes in the standard MIDI format (***M****usical* ***I****nstrument* ***D****igital* ***I****nterface*). For this purpose, we use Eq. (18), where f_i is the frequency in Hz, and x_i is the corresponding MIDI value.

$$x_i = 69 + \frac{12}{\log 2} \log\left(\frac{f_i}{440}\right) \tag{18}$$

B. Variable for Rhythm

Here we wish to relate the variable $y(t)$ of the nonlinear dynamical system to rhythmic values in units of time by using the normalization process $y_n = \gamma(y(t), \mathbf{d})$. This normalization is made with respect to a vector $\mathbf{d}$ of size (1×7) that contains the appropriate numeric values representing the rhythmic notes (a integer number between 6 and 0 is used to represent the interval from the whole note to the sixty-fourth note). In this step, the application of the mapping is unnecessary due to the relationship between the normalized y_n and the rhythmic values, applying the round function. The vectors $\mathbf{y}$ and $\mathbf{x}$ are used to generate the matrix of notes $\mathbf{M}$ and with it the *.mid* file.

The whole process to generate the melody by using a chaotic system is summarized in Fig. 4. It should be noted that only the first and second part of the diagram are used in this paper, which includes all the steps on the left side and center side of the diagram, but not including the steps “Generate Matrix of Notes”and “Write to the MIDI file”.

IV. Musical composition system with LSTM, chaotic inspiration and control strategy

The framework proposed here for music composition has the flexibility to adjust a subjective characteristic of the melody (in this case, the degree of melodiousness), according to a predefined reference value. It is based on the idea that a neural network can learn, in general terms, the characteristics of a melody and these can be complemented by an additional melody named inspiration source. In [10] is shown that controlling the number of notes of the inspiration melody it is possible to vary the pitch-related complexity, the similarity of pitch and rhythm and the degree of originality of the output melody.

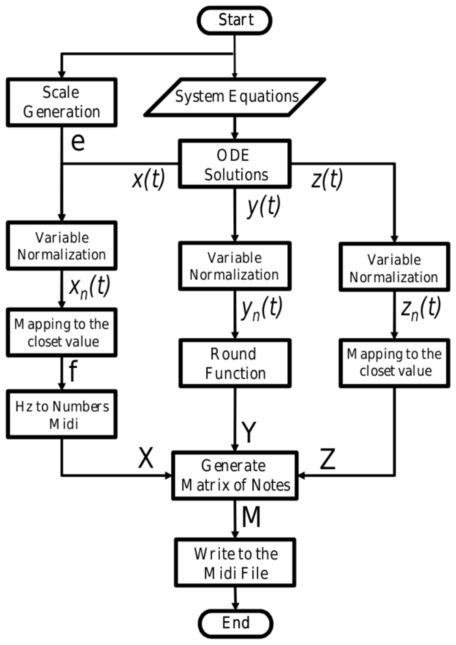

Fig. 4. Diagram of chaotic algorithm [11].

Figure 5 shows the block diagram of the compositional system: r is the reference value of the degree of melodiousness; m is the melodic measure of the output melody; u is the output of the proportional controller, which adjusts the parameter p of the chaotic system; $\mathbf{C}$ is a matrix of solution variables of the chaotic attractor; $\mathbf{M}^{(1)}$ is the melodic matrix of the melody composed by the chaotic musical composition algorithm; $\mathbf{M}^{(2)}$ is the melodic matrix of the input melody; and $\mathbf{M}^{(3)}$ is the melodic matrix of the melody composed by LSTM recurrent neural network.

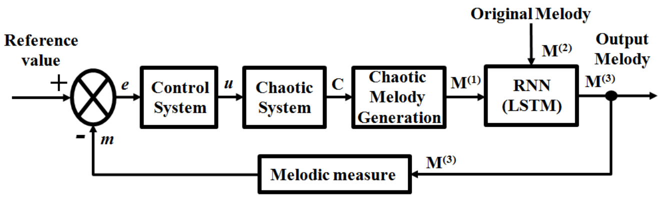

Fig. 5. Diagram of melodic generation system.

The variable error is equal to $e = r - m$. This difference is used in the proportional controller in order to obtain an action u, such that $u = q \cdot e = q \cdot (r - m)$, which changes the parameter of the chaotic system. Therefore the input of the chaotic system f is:

$$\mathbf{C} = f\left(x_0, y_0, p + u\right) \tag{19}$$

where x_0 and y_0 are the initial conditions of the chaotic system and p its parameter. The solution of the system is stored in the matrix $\mathbf{C}_{n\times 2}$ (n is number of iterations). The solution variables are separated, i.e., $x(t) = \mathbf{C}_1$ and $y(t) = \mathbf{C}_2$. Each solution variable of the chaotic system is mapped to a musical aspect, such as pitch and rhythm of the resulting melody. In this case, $x(t)$ is used to create the pitches and $y(t)$ the rhythmic values. Let $\Gamma(\cdot)$ be the chaotic composition algorithm. Then, taking as input the chaotic vectors $x(t)$ and $y(t)$ and the set of musical properties (like scales, number of octaves, initial octave, and so soon) represented with S, the output of chaotic composition block is the melodic matrix $\mathbf{M}^{(1)}$:

$$\mathbf{M}^{(1)} = \Gamma\left(x(t), y(t), S\right) \tag{20}$$

With the melodic matrix of chaotic melody $\mathbf{M}^{(1)}$ the vectors $\mathbf{x}^{(1)} = \mathbf{M}_1^{(1)}$ and $\mathbf{y}^{(1)} = \mathbf{M}_2^{(1)}$ are obtained, which are used as inspiration chaotic inputs that together with the pitches and rhythmic vector of the input melody $\mathbf{x}^{(2)}$ and $\mathbf{y}^{(2)}$ are used with inputs for the LSTM recurrent neural network, obtaining the melodic matrix of the output melody $\mathbf{M}^{(3)}$, as shown in the expression (21):

$$\mathbf{M}^{(3)} = \Omega\left(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}, \mathbf{x}^{(2)}, \mathbf{y}^{(2)}, Q\right) \tag{21}$$

where Ω represents the LSTM network and Q represents the set of input parameters of the network. Finally, let $\theta(\cdot)$ be a function that returns the value of degree of melodiousness of the output melody. Thus $m = \theta\left(\mathbf{M}^{(3)}\right)$, which is the value that will be feedback to the input.

V. Melodic Measures

In this section, the subjective measures to quantify the characteristic of the melody composed by the network and by the chaotic system are described. We adopted the melodic complexity in order to find the relation between one of the parameters of the chaotic system and the generated melody. To quantify the melody generated by the neural network, we use the degree of melodiousness as the target measure.

A. Expectancy-based model of melodic complexity

This model is based on the melodic complexity expectancy, since the components of the model are derived from the melodic expectancy theory. The melodic complexity can be tonal (pitch-related complexity), rhythmic (rhythm-related complexity) or joint (combination of pitch and rhythm-related), where high values indicate high complexity [13].

B. Degree of melodiousness

According to L. Euler (1707-1783), the degree of melodiousness is related to the complexity of mental calculation performed by a listener which is inversely proportional to his pleasant experience [14]. For each interval i ($i \in [1, m]$) of the melody (m is the total number of intervals), a new value a_i is calculated like $a_i = n(i) \times d(i)$, where $\frac{n(i)}{d(i)}$ is the frequency ratio of the current interval. In other words, $n(i)$ and $d(i)$ are the nominator and the denominator of the frequency ratio of the ith interval, respectively. Thus, a_i can be decomposed into products of powers of different prime numbers. In this sense, $a_i = p_1^{k_1} \cdot p_2^{k_2} \cdot \ldots \cdot p_n^{k_n}$, where p_j represents the jth prime number, k_j is the number of appearances of the jth prime number, and n is the largest prime number in a_i. The degree of melodiousness of the interval value a_i and the degree of melodiousness G of the given melody, which is defined as the mean value of $G(a_i)$, are defined by Eq.(22) and (23), respectively.

$$G\left(a_i\right)=\sum_{j=1}^{n}\left(k_j p_j-k_j\right)+1 \tag{22}$$

$$G=\frac{1}{m}\sum_{i=1}^{m}G(a_i) \tag{23}$$

The degree of melodiousness is low if the decomposition contains number primes with low values. On the other hand, it is high if it has number primes with high values and/or it has a lot of prime numbers.

VI. Experimental Results

In this section, we describe the experiments and the results. Our main objective is to automatically compose a melody with a degree of melodiousness predefined. Two LSTM networks, each containing two memory blocks with one memory cell, are used to train pitch and rhythm individually. The network for pitch has 16 inputs and 9 outputs. The first 9 input units correspond to the notes of the input melody, which is represented with 7 bits for pitch and 2 bits for the octave. The remaining 7 input units correspond to the chaotic notes, in the 7-bit representation as well. The network for rhythm has 14 inputs, 7 units for the note of each melody. Both networks were trained with a learning rate equal to 0.1. It was used a discrete chaotic system with two variables, the Hénon map, which is described by the system of equations (24).

$$\begin{aligned} x_{n+1} &= y_{n+1}-ax_n^2 \\ y_{n+1} &= bx_n \end{aligned} \tag{24}$$

where x_n and y_n are the variables of the system, a and b the parameters, and n the number of iterations. Figure 6 illustrates the bifurcation diagram of Hénon map when varying the parameter a between 0 and 1.4 and keeping b fixed at 0.3.

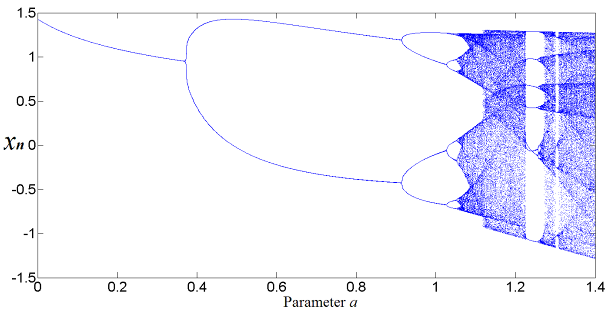

Fig. 6. Bifurcation diagram of Hénon map.

Initially, it is necessary to determine the regions where the pitch-related complexity of the melodies generated by Hénon map is ascendant. Figure 7 shows the pitch-related complexity for chaotic melodies generated by Hénon map varying the parameter a between 0 and 1.4 and with b fixed at 0.3. Figure 8 presents the ascendant region (between 0 and 0.34) to be used in the full system.

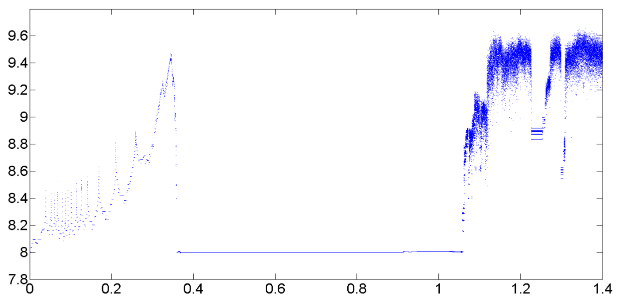

Fig. 7. Pitch-related complexity for melodies generated with Hénon map.

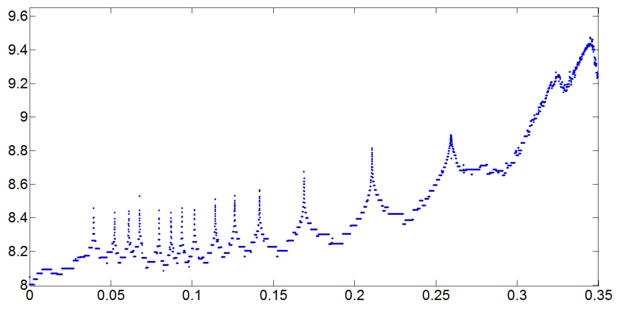

Fig. 8. Ascendant region of pitch-related complexity.

Subsequently, the input melody is chosen. Figure 9 shows the first measures of the piano sonata No. 16 in C major by Mozart. This melody is further used to train the LSTM network.

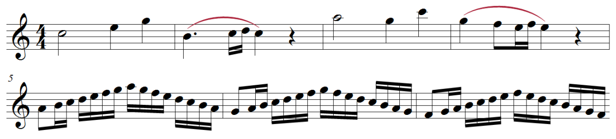

Fig. 9. Measures of the Sonata No. 16 in C major K. 545 by W.A. Mozart.

Figure 10 shows the degree of melodiouness of the melodies composed by LSTM network when varying the parameter a of Hénon map between 0 and 0.4 and with parameter b fixed at 0.3. The dotted line is the degree of melodiouness of the input melody and the solid line is the degree of melodiouness of each one of the composed melodies using different value of parameter a. We can observe small regions where the parameter a is proportional to the melodiouness.

The region between 0.1 and 0.2 covers almost the entire range of melodiouness, i.e., in range of 8 degrees (of 1 through 9). Now the proportional controller is applied with a value of $k = 0.01$ and the initial condition for the parameter a is set equal to 0.1. The compositional system is executed

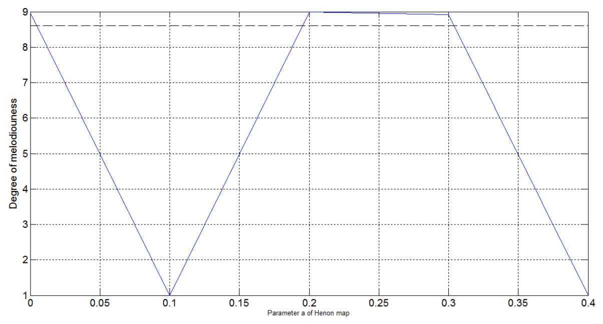

Fig. 10. Degree of melodiouness with parameter a of Hénon map.

to compose a melody with a degree of melodiouness degree equal to 5 with a margin of error of 5%. Figure 11 shows the evolution of parameter a, which reaches the value 5 after 672 iterations.

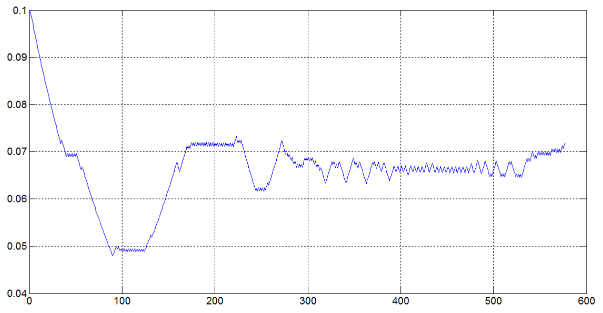

Fig. 11. Evolution of parameter a until it reaches the predetermined value of melodiouness (5).

The melody shown in Fig. 12 was composed by using the full compositional system, this melody has a degree of melodiouness equal to 5.

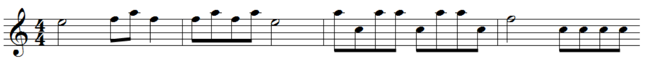

Fig. 12. Composed melody with a degree of melodiouness equal to 5.

VII. Conclusions

In this paper, a LSTM recurrent neural network is used in the context of melody generation with a degree of melodiouness predefined. The LSTM network was modified with an additional input entry that receives a chaotic melody of inspiration. The chaotic inspiration is generated by a chaotic dynamical system through a special algorithm of composition. Taking advantage of the sensitivity to initial conditions and to parameter configuration of the chaotic dynamical systems, several melodies with different complexity degrees can be used as inspiration source. It was verified that the complexity of chaotic melody may be modified by small changes in a parameter of the chaotic system. In this work we have explored the two-dimensional Hénon map, where the first variable was used to generate the pitches and the second the rhythmic durations. The melody composed by the network was quantified using a measure that reflects the degree of melodiouness. We found small regions where there is some degree of proportional relationship between these two variables. With this information, we can use a control strategy that allows the modification of the parameter, aiming at obtaining a melody with a degree of melodiouness that is close to the predefined reference value. These are the first efforts to prove that it is possible to automatically control the subjective characteristics of a melody. However, the system requires several empirical methods and the results are approximate. As future works, we intend to use chaotic systems with three dimensions, with the third variable been capable to generate musical dynamics. It is also proposed to use different types of melodies and different types of chaotic dynamical systems such as fractals and oscillators. In addition, we would like to used other control strategies such as the proportional-integral-derivative controller (PID controller), or, due to the behavior of chaotic systems, a nonlinear control strategy as method OGY, Method of Fixed Point Induction Control (FPIC) or Time-Delayed Feedback (TDAS). Finally, we will also compare the results obtained with LSTM and BPTT, as well as other types of networks.

References

[1] R. Moore, *Elements of Computer Music.* Prentice Hall, 1998.
[2] C. Chen and R. Miikkulainen, "Creating Melodies with Evolving Recurrent Neural Networks," *Proceedings of the Internacional Joint Conference on Neural Networks (IJCNN'01)*, pp.2241-2246, 2001.
[3] B. Pearlmutter, "Gradient calculations for dynamic recurrent neural networks: A survey," *IEEE Transactions on Neural Network,* vol. 6, no. 5, pp.1212-1228, 1995.
[4] S. Hochreiter, Y. Bengio, P. Frasconi and J. Schmidhuber, "Gradient flow in recurrent nets: The difficulty of learning long-term dependencies," *A Field Guide to Dynamical Recurrent Networks, IEEE Press*, 2001.
[5] F. Gers, "Long Short-Term Memory in Recurrent Neural Networks," *Ph.D. thesis, École Polytechnique Fédérale de Lausanne EPFL*, 2001.
[6] D. Eck and J. Schmidhuber, "Finding Temporal Structure in Music: Blues Improvisation with LSTM Recurrent Networks," *Proceedings of the 2002 IEEE Workshop, Networks for Signal Processing XII*, pp.747-756, 2002.
[7] D. Eck and J. Schmidhuber, "Learning the Long-Term Structure of the Blues", *Proceedings of the International Conference on Artificial Neural Networks*, pp.284-289, 2002.
[8] A. Brandmaier, "ALICE: A LSTM-Inspired Composition Experiment," *Thesis (Diplomarbeit) of the Computer Science programme of the Technical University of Munich*, 2002.
[9] D. Côrrea, J. Saito and S. Abib, "Composing Music with BPTT and LSTM Networks: Comparing Learning and Generalization Aspects," *Proceedings of the 2008 11th IEEE International Conference on Computational Science and Engineering (CSEW'08) - Workshops*, pp.95-100, 2008.
[10] A. Coca, F. Roseli and L. Zhao, "Generation of composed musical structures through recurrent neural networks based on chaotic inspiration," *Proceedings of the 2011 International Joint Conference on Neural Networks (IJCNN'11)*, pp.3220-3226, 2011.
[11] A. Coca, G. Olivar and L. Zhao, "Characterizing Chaotic Melodies in Automatic Music Composition," *CHAOS - An Interdisciplinary Journal*, vol. 20, no. 3, pp.033125, 2010.
[12] J. Franklin, "Recurrent Neural Networks for Music Computation," *Journal on Computing*, vol. 18, no. 3, pp.321-338, 2006.
[13] A. Eerola, "Expectancy-based model of melodic complexity," *Proceedings of the Sixth International Conference on Music Perception and Cognition (ICMPC)*, 2000.
[14] L. Euler, *Tentamen Novae Theoriae Musicae Ex Certissismis Harmoniae Principiis Dilucide Expositae.* Saint Petersburg Academy, 1739.