2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

# Detection of filter-like cellular automata spectra

Eurico L.P. Ruivo and Pedro P.B. de Oliveira

***Abstract*—The Fourier spectra of one-dimensional cellular automata give a quantitative and qualitative characterisation of the average final configurations obtained out of their rules, as they are applied to sets of random initial configurations. The elementary cellular automata rule space presents spectra that bring to mind those of digital filters, and the same happens to some of the cellular automata rules obtained through composition of particular elementary cellular automata. As such, one might be willing to discover other filter type rules that might exist in larger spaces. In order to explore the possibility of detecting these cellular automata in a larger space, two methods are applied: a Multilayer Perceptron and the k-Nearest Neighbours classification algorithm. Both algorithms presented considerably high accuracies, with the Multilayer Perceptron showing an overall lower false negative rate, thus indicating that the methods may be generalised to other rule spaces and to the detection of other features, providing an automatic method to detect features in cellular automata spectra.**

## I. Introduction

Cellular automata (CAs) are fully discrete dynamical systems, in space, time and state variables, based upon a homogeneous, parallel and locally-defined operation. Some well-known simple CAs, such as the elementary cellular automaton (ECA) *rule 110* and John Conway's *Game of Life*, present computational universality, thus showing that, despite their simplicity, they are capable of giving rise to complex behaviour. From another viewpoint, CAs can also be used as models to simulate real systems and complex phenomena ([1]), including social systems, disease propagation ([2]), fluid dynamics ([3]) and many others.

From the conjecture that rules with universal computability present 1/f noise in their spectra, [4] applied genetic algorithms to search for rules with such a spectral behaviour, concluding that the fittest rules found by the algorithm display propagating structures in their time evolution, which is a feature of computationally universal rules.

Not only computational universality can be studied from the viewpoint of the spectral analysis of CAs; for instance, [5] correlates the spatial patterns and attractors found in the time evolution of CA rules with their *power spectra*, that is, the Fourier spectra squared coordinate-wise. Also, [6] applied spectral analysis to evaluate a particular rule in a CA-based computational task (the parity problem) for configurations of a given length.

The spectrum of a CA rule gives statistical information on its dynamical behaviour ([7]). One of the problems proposed in [7] is to investigate the properties of the *limit sets* of CA rules. A limit set (or a *$\omega$-limit set*) of a cellular automaton is the set of configurations which are not excluded during the time evolution of the rule. At a finite time span, the set of configurations that a CA rule can generate can always be described by a regular language, and that is the case for the limit set of some particular rules [8], as other rules may have limit sets that can only be described by context-free, context-sensitive or recursively enumerable languages. It has been shown that it is undecidable whether two CAs display the same limit set, or whether a limit set can be described by a regular language ([9]), the latter only being decidable for particular cases, as happens in [10], where ECA rule 22 is shown to have a limit set that cannot be described by a regular language.

The analysis of CAs from the viewpoint of their spectral signature provides statistical information on their limit behaviour ([7]). The *Fourier spectra* described in this paper give an average representation of the distinct patterns that can be found in configurations obtained from the iteration of a CA rule, over a set of initial configurations, after a transient time. Such representations are given in terms of a frequency domain ([11]).

Since even simple CAs have been used to filtering tasks in image processing ([12], [13], [14]), it is natural to ask whether rules applied as filters present filter-like spectra or, conversely, whether rules that present filter-like spectra could be, somehow, used as filters. Here we focus on the use of two techniques applied to the detection of filter-like cellular automata spectra, namely the ***Multi**Layer **P**erceptron* (MLP) neural network and the ***k** **N**earest **N**eighbours* (kNN) classification algorithm.

The number of rules, and thus the number of rule spectra, increases exponentially with the size of the neighbourhood and the number of states of the CA, which renders manual classification unfeasible for large spaces. Here we present two methods for obtaining

E.L.P. Ruivo (euricoruivo@gmail.com) and P.P.B. de Oliveira (pedrob@mackenzie.br) belong to the Postgraduate Programme in Electrical Engineering, from Mackenzie Presbyterian University, So Paulo, SP, Brazil, and P.P.B.O. is also a member of its School of Computing and Informatics.

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

automatic classification of the spectra using a relatively small amount of rules of a given rule space.

The *Multilayer Perceptron* (MLP) provides a framework to address non-linearly separable problems, as well as a standard method for classification and pattern recognition used broadly for those tasks. Also, the MLP has the ability to generalise, what is particularly important for the issue of obtaining classifications in terms of spectral features for larger spaces.

Signal detection and/or classification problems have been addressed by means of MLP networks. In [15], an MLP is applied to the detection and classification of orthogonal signals in environments with and without noise and compared against an optimal classifier, with the MLP classifier converging to an optimal performance for noise-free environments. MLPs also have relevant applications in real-world problems, such as classification of electroencephalograph (EEG) ([16], [17]) and electrocardiogram (ECG) ([18]) signals.

Since MLPs are applied to both theoretical and real-world signal classification problems, and since CA spectra can be regarded as signals, the MLP structure is suitable to the proposed problem of filter-like spectra detection.

The *k Nearest Neighbours* (k-NN) algorithm ([19]) is a very straightforward nonparametric method for classifying objects in a database. In order to classify an (unclassified) object, one assigns to it the most common class among its k classified neighbours. Also, it has been largely used for many different classification and pattern recognition tasks. However, the main purpose of choosing k-NN here is to see how MLP's performance compares to the performance of a relatively simple and reliable algorithm, in the sense that it was expected to have acceptable performance results without feature extraction of the spectral vectors.

Other algorithms could be taken instead, such as Naïve-Bayes, Particle Swarm Classifier (PSC) or the k-Means clustering algorithm (with the classification induced by the clustering).

Naïve-Bayes and PSC are amenable to solving filter-like spectra detection and other spectral classification problems, such as the presence/absence of energy peaks; however, given the dimension of the search space, instead of dealing with the raw data they would demand a proper choice of features to get extracted from the spectral vectors in order to lead to a good performance. As for the k-Means algorithm, it could be computed without the need to consider a set of parameters in order to reduce the dimensionality of the space, but its sensibility to initialisation would have to be addressed, which – generally speaking – would strongly depend upon which spectral feature there is to be detected/classified.

Also, classifications obtained by PSC or the k-Means algorithm would apply only to the search space considered and could hardly be generalised to higher-dimensional CA rule spaces.

This paper is organised as follows: Section II provides the basic required definitions of cellular automata, as well as the definition of rule composition, the computation of the spectrum, and the spectral feature to be detected. Then, in Section III the methodology used to select the parameters is discussed, together with the training and implementation of the MLP and the k-NN. The analysis of the results obtained is made in Section IV and the concluding remarks are made in Section V

## II. Cellular Automata

### A. Basic definitions

A *cellular automaton* (CA) is a quadruple $A = (S, N, f, d)$, where $S \subset \mathbb{Z}$ is its *state set*, $N \subset \mathbb{Z}^d$ is its *neighbourhood vector*, $f : S^{|N|} \longrightarrow \mathbb{S}$ is its *local transition rule* and $d \in \mathbb{Z}^+$ is its *dimension*.

Put in an intuitive way, a CA acts locally upon a $d$-dimensional discrete lattice of cells, that take on values over the state set. This local update of each cell is given in terms of the neighbourhood vector and the local transition function.

The neighbourhood vector defines which cells directly affect the behaviour of each other. For instance, in a one-dimensional lattice, the neighbourhood vector -1,0,+1 represents that the behaviour of a cell will be affected by the states of its left (-1) and right (+1) neighbours, and by its own state (0).

In the particular case where $d = 1$, the CA is said to be *one-dimensional*. A *radius-$r$ cellular automaton* ($r \in \mathbb{N}$) has the neighbourhood vector system given by $N = N_r = \{i \in \mathbb{Z} : |i| \leq r\}$.

A *binary cellular automaton* has state set $S = \{0, 1\}$. A one-dimensional, radius-1, binary cellular automaton is called an *elementary cellular automaton* (ECA). From now on we only consider one-dimensional, binary, cellular automata.

Formally, an ECA is a quadruple $A = (0, 1, -1, 0, 1, f, 1)$, with $f : S^{|N|} \longrightarrow \mathbb{S}$. Intuitively, an ECA is a local binary (S=$\{0,1\}$) function that acts on a one-dimensional ($d = 1$) array of cells, with each individual cell being updated based upon its left neighbour's state, its own state and its right neighbour's state (N=$\{-1,0,1\}$).

Any binary, one-dimensional radius-r CA $A = (\{0, 1\}, N_r, f, r)$ can be identified by its *Wolfram number* ([1]), $W_A$, given by

$$W_A = \sum_{(a_1,\cdots,a_m)\in 2^m} f(a_1,\cdots,a_m)2^{\sum_{i=1}^{m} a_i 2^{m-i}} \quad (1)$$

where $m = 2r + 1$.

In the remainder of this paper, a CA $A$ will be identified by $A = (S, N_r, f_{W(S,r,f)}, 1)$ or, when it is clear from the context, simply by its Wolfram number. Hence, for instance, the ECA with Wolfram number 110 will be referred to as $(\{0, 1\}, N_1, f_{110}, 1)$ or as *rule 110*.

A *configuration* $c$ is a function $c : \mathbb{Z} \longrightarrow S$, that associate to every integer an element of the state set. The set of all configurations over the state set S will be denoted as $X(S)$.

A configuration is said to be under *periodic boundary condition* (PBC) if there is $p \in \mathbb{N}^+$ such that $c(i + p) = c(i), \forall i \in \mathbb{Z}$. One can read this as if the cells were distributed over a one-dimensional torus. Configurations under PBC are also referred to as *spatially periodic configurations*.

Given a radius-$r$ CA with local transition rule $f$, a global transition function $F : X(S) \longrightarrow X(S)$ is induced by $f$, given by:

$$[F(c)](i) = f(c(i - r), \cdots, c(i + r)), i \in \mathbb{Z} \quad (2)$$

The iteration of $F$ $t$ times ($t \geq 0$) over a configuration $c$ is denoted by $F^t(c)$. For the sake of simplicity, we will define $F^0(c) = c$. The *time evolution* of a CA $A = (S, N, f, 1)$ *from a configuration* $c_0$ is the set

$$E_f^{c_0} = \{F^t(c_0) : t \in \mathbb{Z}, t \geq 0\} \quad (3)$$

It is worth noticing that the image of a PBC configuration by a global rule F is also a PBC configuration. Hence, the set $E_f^{c_0}$ contains only PBC configurations, given that $c_0$ is under PBC.

The PBC configurations are used to an effective simulation of a CA rule in a computer [20]. For this reason, the Fourier spectrum of a CA described in Section II-C are computed over sets of PBC configurations obtained from a finite time evolution, with the time taken sufficiently large in order to avoid any transient behaviour.

### B. Rule composition

As described in [21] and [22], given two one-dimensional CAs $A = (S, N_{r_A}, f_A, 1)$ e $B = (S, N_{r_B}, f_B, 1)$, it is possible to compute the *composition of* $A$ *and* $B$, $A \circ B = (S, N_{r_A+r_B}, f_A \circ f_B, 1)$, where

$$\begin{aligned}[(f_A \circ f_B)(c)](i) = \quad & f_A(f_B(i - r_A - r_B, \cdots, i - r_A + r_B),\\ & \cdots, f_B(i + r_A - r_B, \cdots, i + r_A + r_B)),\\ & i \in \mathbb{Z}, c \in X(S)\end{aligned}$$

Hence, taking local rules $f \doteq f_{54}$ and $g \doteq f_{110}$ (both featuring a complex dynamical behaviour), it is possible to compute a radius-2 rule $h \doteq (f \circ g)$. For instance, $h(0, 1, 1, 0, 1)$ is given by

$$\begin{aligned}&h(0, 1, 1, 0, 1) =\\ &= f(g(0, 1, 1), g(1, 1, 0), g(1, 0, 1)) = f(1, 1, 1) = 0\end{aligned}$$

The composition of CAs is defined as described above in a way such that the time evolution of the composition is the same time evolution obtained by applying $f_B$ and $f_A$ alternately to the configurations, that is, $E^{c_0}_{(f_A \circ f_B)} = \{c_i : i \in \mathbb{N}\}$, where $c_i = F_A(F_B(c_{i-1})), i \in \mathbb{N}, i \geq 1$. Notice that the composition operation is not commutative. Hence, in general, $A \circ B \neq B \circ A$.

The diversity of dynamical behaviours found in CA rule spaces increases with the value of the radius. Therefore, since the composition of two rules yields a rule with larger radius, consequently, the set of rules obtained by composition present more diverse dynamical behaviours.

### C. Fourier Spectra

Given a n-dimensional complex vector $w \in \mathbb{C}^n$, the *discrete Fourier transform* of $w$, $\mathscr{F}(w)$, is given by

$$[\mathscr{F}(w)](k) = \frac{1}{n}\sum_{j=1}^{n} u_j e^{2\pi \mathrm{i}(j-1)(k-1)/n} \quad (4)$$

Given a radius-$r$ CA $A = (S, N_r, f, 1)$, its spectrum, $\mathscr{S}_A$, is given by:

$$\mathscr{S}_A = \frac{1}{N}\sum_{c\in C} |\mathscr{F}((f_\infty)^t(c))| \quad (5)$$

In the above equation $C$ is a set of random initial configurations, $N = \#C$, $t \in \mathbb{N}$ is the number of time steps taken sufficiently large in order to avoid transient behaviour, as discussed in Section II, and $|\mathscr{F}(\cdot)|$ is the absolute value of the Fourier transform taken coordinate-wise.

The Fourier transform transposes a sequence to its frequency domain. Therefore, the Fourier spectrum of a CA described above contains information, in the frequency domain, concerning the average configuration obtained in the end of the time evolution. In the case of the binary CAs addressed here, the frequency at issue corresponds to *bit* changes in a binary sequence. For instance, while the sequence $\cdots 101010 \cdots$ has maximal frequency, the sequences $\cdots 000000 \cdots$ and $\cdots 111111 \cdots$ have minimal frequencies. All the other possible sequences present intermediate frequencies, depending on the frequency of *bit* changes. Hence, we can refer to an *energy* level of each frequency that indicates, in average, the extent in which the blocks of a frequency are present relatively to the others [11].

The ECA rule 184 (known as the *"traffic rule"*) provides a very illustrative example. The configurations obtained after a sufficiently large number of time steps can be described in terms of the relative frequency of 1s in the initial configuration, as follows: if there is the same number of 0s and 1s, two consecutive cells present distinct states (that is, the final configuration is $\cdots 101010 \cdots$), which is a sequence with maximal frequency; if there are more 0s than 1s, the final configuration displays alternating 0s and 1s, the exceeding 0s giving rise to clusters of 0s; and if there are more 1s than 0s, the final configuration obtained is analogous to the previous case. Therefore, starting with configurations with more 0s than 1s, for instance, and going to configurations with the same number of 0s and 1s, a decrease of the energy in the lower frequencies is observed, with a corresponding increase on the energy levels of the higher frequencies, up to the point when only the maximal frequency is represented, as shown in Figure 1.

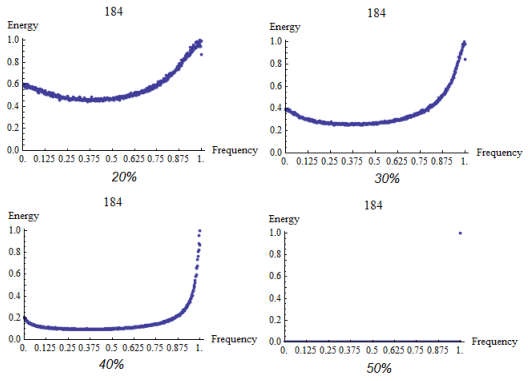


Fig. 1. Spectra of the ECA rule 184 for sets of initial configurations with distinct relative frequencies of 0s and 1s. The percentage below each graph indicates the relative frequency of 1s in each initial configuration of the set used to compute the spectrum in each case.

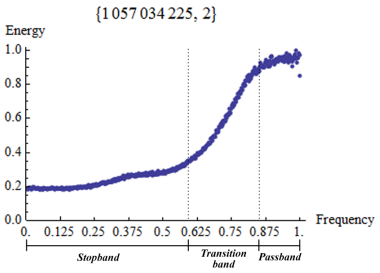


Fig. 2. Spectrum of radius-2 rule 1,057,034,225 as a high-pass filter.

### D. Filter-like spectra

Cellular automata and system based on them have been used in many tasks of image processing. In [12] a two-dimensional CA was applied to noise-filtering in images with different levels of noise and one of the models entailed better performance than traditional filters, such as the median filter. Also, methods to obtain CAs able to perform image processing tasks, such as noise-filtering, border and connected component detection, are described in [13].

A particular family of CAs, the *Multiple Attractors Cellular Automata* (MACA), is used in [14] to build an evolutionary based classification system. The resulting CA is then used to perform image compression, among other applications.

The possibility of using CAs to filter, for instance, brings to mind a particular type of spectrum that some CAs display: the filter-like spectra. An example is the spectrum of radius-2 rule 1,057,034,225 (Figure 2), that resembles a high-pass filter spectrum, with low energy in the lower frequencies, a transition band of frequencies and a *plateau* of high energy levels in the higher frequencies. Other rules have a spectrum that resembles a low-pass filter (high energy levels in low frequencies and low energy levels in high frequencies) and, of course, there are rules with spectra that do not resemble a filter spectrum in any way.

In this paper, the space of one-dimensional radius-2 binary CAs obtained by composing two ECAs is explored using MLP and the k-NN algorithm, in order to search for three kinds os spectra, classified accordingly to their similarity to filters: *non-filter spectra* (NF), *low-pass spectra* (LP) and *high-pass spectra* (HP)

## III. Methodology

### A. The spectral database

The number of binary, one-dimensional, radius-$r$ CAs increases exponentially in respect to $r$. For $r = 1, r = 2$ and $r = 3$ there are, respectively, $2^8, 2^{32}$ and $2^{128}$ distinct CA rules. Therefore, in order to classify

each rule according to its spectrum, it is necessary to apply a reliable automatic procedure. Here, the automatic classification of a subset of the radius-2 rules is obtained, by using a manually classified random sample of the spectra. The idea is both to give an efficient method of filter-like spectra detection for more general rule spaces, and to investigate what kind of spectral features can arise from CA rule composition.

In order to compute the spectra, each rule was applied to a set of 1,000 periodic configurations of 1,024 bits for 200 time steps, giving rise to a 512-dimensional real vector, namely, the rule spectrum. The set of parameters above is the same one used in [11] to partition the ECA rule space into spectral classes.

Two CA rules are said to be dependent if they have the same dynamical behaviour, up to permuting the states and/or mirroring the configurations; otherwise, the rules are said to be independent. Hence, in order to study a rule space, it suffices to study the set of its independent rules.

By taking the 88 dinamically independent ECA rules ([1]) and composing them, a set of 5,766 radius-2 rules are obtained. Hence, the database consists of 5,766 real vectors with 512 coordinates. Each coordinate is normalised by the maximum energy level of the spectrum, as in [11].

In order to classify the spectra, a training set of 500 (approximately 8.7% of the spectra database) was randomly chosen from the database of 5,766 ECA-derived rules, in a way that tried to preserve in it the same proportion of the three classes, as they appear in the entire database.

### B. The Multilayer Perceptron implementation

A *Multilayer Perceptron* (MLP) was trained with the spectra in the training set in order to classify the remaining spectra. At first, the *logistic sigmoid* was used as the activation function of each neuron, but that condition was later relaxed, by means of variations of that classical approach, in order to obtain a smaller *Mean Square Error* (MSE) for the training. The weights were updated using backpropagation and the network had one hidden layer.

The MLP was trained for different values of the number of neurons in the hidden layer ($N_h$) and for the learning rate ($\alpha$).

In order to observe the influence of $N_h$ in the learning process, the value $\alpha = 0.1$ was fixed and $N_h$ was taken in the set $\{4, 8, 16, 32\}$, and the MSE variation along the first 100 epochs was analysed. The results of this test are shown in Figure 3. Powers of two were chosen for the values of $N_h$ since the number

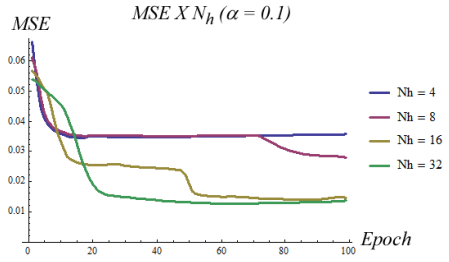


Fig. 3. MSE variation along the first 100 epochs for distinct values of $N_h$.

of inputs is also a power of two (each spectrum is a 512-dimensional vector).

The fast decrease of the MSE and its subsequent stabilisation in the first epochs for $N_h = 4$ and $N_h = 8$ shows that the net inputs of the hidden layer neurons fall out of the more dynamically active portion of the activation function domain (logistic sigmoid). A change in the activation function was later used in order to tackle this issue, as described later on. Nevertheless, for $N_h = 8$, around the 70th epoch, MSE diminishes again, indicating that the system was able to keep learning despite the stabilisation of MSE in the first few epochs.

For $N_h = 16$ and $N_h = 32$, the MSEs obtained by the end of the 100th epoch are statistically equivalent, with the MSE decay for $N_h$ occurring in a step like fashion; for $N_h = 32$ the decay is smooth at first, becoming more accentuated around the 20th epoch. It is worth noticing that the minimal MSE for $N_h = 32$ was obtained before the 100th epoch, around the 60th and 65th epoch.

Then, taking $N_h = 32$, the learning rate $\alpha$ was taken in $[0.1, 1.0]$ with step-size 0.1. The result is shown in Figure 4. While for $\alpha \in [0.2, 0.5]$ the MSEs decrease sharply in the first epochs, displaying a noisy variation at the final epochs, for $\alpha = 0.1$ the decay is smoother and more stable, and does not lend itself to noisy variation in the final epochs. With $\alpha \in [0.6, 1.0]$ the MSE variation in a few epochs become progressively larger with the increase of $\alpha$.

Since for $\alpha = 0.1$ the MSE presents a more stable and smoother behaviour along the epochs, this value was taken, along with $N_h = 32$, towards the last step in setting the parameters for training the network: the use of different activation functions. A family $\{f_\lambda\}$ of variants of the logistic sigmoid was taken, with

$$f_\lambda(x) = \frac{1}{1 + e^{-\lambda x}}, x \in \mathbb{R} \qquad (6)$$

Notice that $f_1$ is the logistic sigmoid itself. The

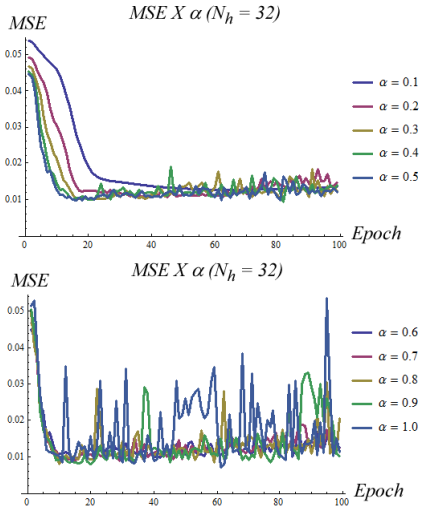


Fig. 4. MSE variation along 100 epochs for $\alpha \in [0.1, 1.0]$. For $\alpha$ between 0.2 and 0.5, the MSEs decay rapidly and then begin to present small noisy variations. With $\alpha = 0.1$ a smoother behaviour is observed. For $\alpha \geq 0.6$, the MSEs display greater variations, increasing with $\alpha$

$\lambda$ parameter was taken between 1.0 e 2.0, with step size 0.25 and, as done for the other parameters, the MSE variation was analysed along 100 epochs. Figure 5 depicts the results.

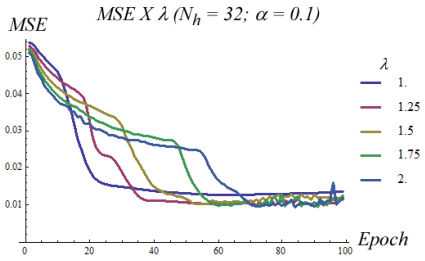


Fig. 5. MSE variation for 100 epochs for distinct values of $\lambda$.

Around the 30th epoch, the MSE for $\lambda = 1.25$ is minimal. Between the 50th and 70th epochs, the MSEs for $\lambda = 1.25$ and $\lambda = -1.5$ become equivalent; however, in the last few training epochs, the MSE for $\lambda = -1.5$ oscilates and grows, which does not occur for $\lambda = -1.25$.

Hence, an MLP with two layers, 32 neurons in the hidden layer, learning rate $\alpha = 0.1$ and activation function $f_\lambda(x) = \frac{1}{1+e^{-1.25x}}$ was trained for 500 epochs in order to classify the aforementioned 5,766 spectra (both the training set and the unclassified spectra set) in three classes: *non-filter* (NF), *low-pass filter* (LP) and *high-pass filter* (HP).

The results obtained are discussed in Section IV.

### C. The k Nearest Neighbours algorithm

For $k \in \{1, 3, 5, 7, 9\}$, the k-NN algorithm was applied to determine whether a spectrum was a non-filter, a low-pass filter or a high-pass filter, as follows:

- While there is an unclassified spectrum, do:
  - Randomly choose an unclassified spectrum $S_f$;
  - Consider the classes $\{C_1, \cdots, C_k\}$ of the $k$ nearest (in terms of euclidean distance) classified spectra to $S_f$;
  - Assign the mode of $\{C_1, \cdots, C_k\}$ as the class of $S_f$. If $\{C_1, \cdots, C_k\}$ is a multimodal set, randomly assign one of the modes;
- Return the classified spectra.

It should be noticed that k-NN was applied in a semi-supervised fashion: at each iteration, the set of already classified spectra is increased by one and this newly classified spectra is also used in the next iterations to classify the remaining spectra.

For $k \in \{1, 3, 5, 7, 9\}$, the k-NN algorithm was subjected to a process of cross-validation in 10-folds using the same 500 previously classified spectra used for the MLP, where the spectra were distributed among the folds in order to optimise the proportion of NF, LP and HP spectra found in the random sample of the database. Table I shows the performance of the algorithm in terms of the mean accuracy and a mean *false negative rate* (FNR), with the positive "class" being the set obtained by joining LP and HP spectra. More precisely,

$$FNR = \frac{\text{number of LP or HP spectra classified as NF}}{\text{total amount of LP and HP spectra}} \quad (7)$$

Informally speaking, FNR measures measures how often the algorithm "miss" a filter spectrum.

TABLE I. ACCURACY AND FALSE NEGATIVE RATE FOR DISTINCT VALUES OF $k$ IN THE 10-FOLDS CROSS-VALIDATION.

| k | Accuracy | FNR |
|---|---|---|
| **1** | (93.20 ± 3.42)% | (26.00 ± 1.41)% |
| **3** | (94.80 ± 3.01)% | (22.00 ± 7.07)% |
| **5** | (93.40 ± 2.67)% | (24.00 ± 2.83)% |
| **7** | (94.00 ± 2.67)% | (22.00 ± 1.41)% |
| **9** | (93.40 ± 2.32)% | (22.00 ± 1.41)% |

The accuracy of the k-NN is statistically the same for every value of $k$. As for FNR, the minimal mean values were obtained for $k \in \{3, 7, 9\}$ with the minimal standard deviation being achieved for $k \in \{7, 9\}$. Hence, $k = 7$ and $k = 9$ were chosen to apply the algorithm to the whole database.

## IV. Results

The MLP trained to perform detection and classification of filter-like CA spectra led to an accuracy of 92.30% over the database of 5,766 spectra. Table II shows the confusion matrix, accuracy and FNR related to the classification obtained by the MLP.

TABLE II. Confusion matrix of the classification of the 5,766 spectra obtained by the MLP.

| Actual Class | Predicted Class | | |
|---|---|---|---|
| | **NF** | **LP** | **HP** |
| **NF** | 4,822 | 266 | 119 |
| **LP** | 11 | 216 | 0 |
| **HP** | 48 | 0 | 284 |
| Accuracy: 92.30%; FNR: 10.55% | | | |

In terms of absolute values, of the 444 wrongly classified spectra, 385 (86.7%) are due to non-filter spectra being classified as low-pass or high-pass filter spectra. Since the goal was to build an MLP capable of detecting filters, this kind of classification error is not as serious as it would be if the majority of classification errors were due to filter-like spectra being classified as non-filters, what would cause a larger number of filter-spectra to be missed by the network.

Also, since LP and HP spectra are very distinct, from the viewpoint of the energy level distribution among the frequency bands, there was no wrong classification among these classes.

As for k-NN, since it is a simple algorithm and depends only upon $k$, it was possible to obtain complete classifications of the database for distinct values of $k$. Table III shows the confusion matrices, the accuracy and the FNR of the algorithm for $k \in \{7, 9\}$.

TABLE III. Confusion matrices, accuracy and FNR for distinct values of $k$.

| $k = 7$ | **Predicted Class** | | |
|---|---|---|---|
| **Actual Class** | **NF** | **LP** | **HP** |
| **NF** | 5065 | 76 | 66 |
| **LP** | 74 | 153 | 0 |
| **HP** | 64 | 0 | 268 |
| Accuracy: 95.14%; FNR: 24.69% | | | |
| $k = 9$ | **Predicted Class** | | |
| **Actual Class** | **NF** | **LP** | **HP** |
| **NF** | 5056 | 77 | 74 |
| **LP** | 75 | 152 | 0 |
| **HP** | 66 | 0 | 266 |
| Accuracy: 94.93%; FNR: 25.22% | | | |

The k-NN algorithm led to an accuracy around 95% in both cases; however both FNRs are considerably high, as if the algorithm had missed, in average, 1 for each group of 4 filter-like (LP or HP) spectra. Table IV summarises a comparison between MLP and the two variations of k-NN in terms of accuracy, FNR and percentages of misclassification per class (MPC).

TABLE IV. Accuracy, FNR and percentages of missclassification per class for the different algorithms.

| **Algorithm** | **Acc** | **FNR** | **MPC** | | |
|---|---|---|---|---|---|
| | | | **NF** | **LP** | **HP** |
| **MLP** | 92.30% | 10.55% | 7.40% | 4.85% | 14.46% |
| **7-NN** | 95.14% | 24.69% | 2.73% | 32.60% | 19.28% |
| **9-NN** | 94.93% | 25.22% | 2.90% | 33.04% | 19.88% |

Figure 6 shows some examples of LP and HP spectra found by both the MLP and the k-NN in the database of spectra.

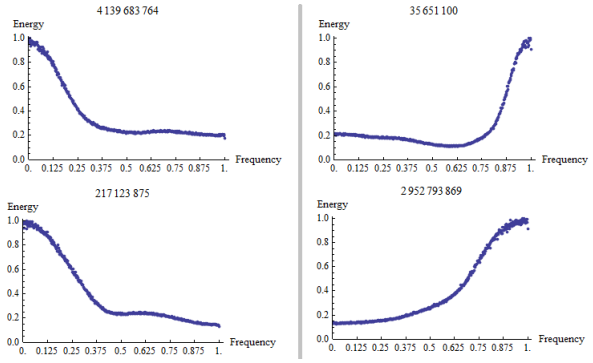


Fig. 6. Examples of low-pass *(left)* and high-pass *(right)* spectra found both by the MLP and the k-NN in the database.

Both k-NN have shown slightly higger accuracy than that of MLP; however, MLP has outperformed k-NN in terms of FNR, having less than half the amount of filters that have been missed by the k-NN. Also, looking at the percentage of misclassifications per class (MPCs), MLP has the lowest percentages for LP and HP, worth noticing being that the MPC of the LP class is roughly 8 times lower than these for the k-NNs. As for the MPC of the NF class, the k-NNs outperformed MLP with percentages about 2.5 times lower than that of the MLP.

The data in Table IV shows that MLP had a better performance in filter detection (particularly filters of the LP class), and k-NN had a better performance detecting non-filters, thus suggesting that an ensemble using both methods may lead to even better results, certainly a theme for further research.

## V. Concluding Remarks

This paper presented the training and the application of a Multilayer Perceptron and the k Nearest Neighbours classification algorithm to the detection of

filter-like cellular automata spectra. While both algorithms had high accuracy values, the MLP has shown better performance in terms of not mistaking filters (LP and HP) for non-filters (NF), which is more relevant to the detection of filters than doing the converse, that is, not mistaking non-filters for filters.

As mentioned before, manual classification of the spectra of large rule spaces is unfeasible, since the number of spectra grows exponentially with the size of the neighbourhood and the number of states, while the procedures described here provides a way to automatically classify spectra of larger spaces by using just a relatively small number of rule spectra.

The intrinsic generalisation ability of the MLP brings the possibility to use the model trained in a rule space of an arbitrary dimension to search for filter-like spectra in more general, larger CA rule spaces.

Filter-like spectra suggest that cellular automata rules associated with them might be used as different kinds of filters, depending upon the dynamical behaviour of each particular rule. However, it is necessary to search for spectra with that feature in more general rule spaces; and this must be done in an efficient, accurate and automatic way, based upon a relatively small sampling of the whole space. The application of other models and methods to the problem tackled herein is certainly appealing as a follow-up, such as semi-supervised learning and ensemble methods. Equally important would be the individual analysis of rules applied to sets of initial configurations, so as to evaluate the possibility of using them as effective filters.

Further research topics include searching more general rule spaces for rules that can, in a practical and useful way, be applied as filters.

Conversely, it is also useful to find filter based CAs by means of dynamical analysis and then compute their spectra, which could validate the correlation between filter-like spectra and *filter-behaviour* of CA rules.

## ACKNOWLEDGEMENTS

We thank the continued support to our research, provided by the Brazilian agencies CAPES, FAPESP and MackPesquisa.

## REFERENCES

[1] S. Wolfram, *A New Kind of Science*. Wolfram Media, 2002.

[2] G. Sirakoulis, I. Karafyllidis, and A. Thanailakis, "A cellular automaton model for the effects of population movement and vaccination on epidemic propagation," *Ecological Modelling*, vol. 133, no. 3, pp. 209–223, 2000.

[3] N. Margolus, T. Toffoli, and G. Vichniac, "Cellular-automata supercomputers for fluid-dynamics modeling," *Physical Review Letters*, vol. v56, no. n16, p. p1694, 1986.

[4] S. Ninagawa, "Evolution of one-dimensional cellular automata by 1/ f noise," *Spectrum*, vol. 4648, pp. 905–914, 2007.

[5] W. Li, "Power spectra of regular languages and cellular automata," *Complex Systems*, vol. 1, pp. 107–130, 1987.

[6] S. Ninagawa, "Solving the parity problem with elementary cellular automaton rule 60," *Proceedings of AUTOMATA 2012: 18th International Workshop on Cellular Automata and Discrete Complex Systems*, 2012.

[7] S. Wolfram, "Twenty problems in the theory of cellular automata," *Physica Scripta*, vol. T9, no. 3, pp. 170–183, 1985.

[8] ——, "Computation theory of cellular automata," *Communications in Mathematical Physics*, vol. 1, no. 96, pp. 15–57, 1984.

[9] L. Hurd, "Formal language characterizations of cellular automaton limit sets," *Complex Systems*, vol. 1, no. 1, pp. 69–80, 1987.

[10] Z. Jiang and Y. Wang, "Complexity of limit language of cellular automaton rule 22," *Applied Mathematics – A Journal of Chinese Universities*, no. 20, pp. 268–276, 2005.

[11] E. L. P. Ruivo and P. P. B. de Oliveira, "A spectral portrait of the elementary cellular automata rule space," in *Irreducibility and Computational Equivalence*, ser. Emergence, Complexity and Computation, H. Zenil, Ed. Springer Berlin Heidelberg, 2013, vol. 2, pp. 211–235.

[12] P. J. Selvapeter and W. Hordijk, "Cellular automata for image noise filtering," pp. 1–5, 2009.

[13] P. L. Rosin, "Image processing using 3-state cellular automata," *Computer Vision and Image Understanding*, vol. 114, no. 7, pp. 790–802, 2010.

[14] P. Maji and C. Shaw, "Theory and application of cellular automata for pattern classification," *Performance Computing*, vol. 58, no. 3, pp. 321–354, 2003.

[15] Z. H. Michalopoulou, L. Nolte, and D. Alexandrou, "Performance evaluation of multilayer perceptrons in signal detection and classification," *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 381–386, 1995.

[16] E. Haselsteiner and G. Pfurtscheller, "Using time-dependent neural networks for eeg classification," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 4, pp. 457–463, 2000.

[17] "EEG signals classification using the k-means clustering and a multilayer perceptron neural network model," *Expert Systems with Applications*, vol. 38, no. 10, pp. 13 475 – 13 481, 2011.

[18] V. Mai, I. Khalil, and C. Meli, "ECG biometric using multilayer perceptron and radial basis function neural networks," in *Engineering in Medicine and Biology Society,EMBC, 2011 Annual International Conference of the IEEE*, 2011, pp. 2745–2748.

[19] T. Cover and P. Hart, "Nearest neighbor pattern classification," *Information Theory, IEEE Transactions on*, vol. 13, no. 1, pp. 21–27, 1967.

[20] J. Kari, "Theory of cellular automata: A survey," *Theoretical Computer Science*, vol. 334, no. 1-3, pp. 3–33, 2005.

[21] G. A. Hedlund, "Endomorphisms and automorphisms of the shift dynamical system," *Mathematical Systems Theory*, vol. 3, no. 4, pp. 320–375, 1969.

[22] A. Schranko and P. P. B. de Oliveira, "Derivation and representation of one-dimensional, reversible, number-conserving cellular automata rules," *Journal of cellular automata*, pp. 77–89, 2010.