Neurocomputing ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Cloning DRASiW systems via memory transfer

Massimo De Gregorio [a], Maurizio Giordano [b,*]

[a] Istituto di Scienze Applicate e Sistemi Intelligenti CNR, Via Campi Flegrei, 34 - 80078 Pozzuoli, Naples, Italy
[b] Istituto di Calcolo e Reti ad Alte Prestazioni CNR, Via Pietro Castellino, 111 - 80131 Naples, Italy

ARTICLE INFO

Article history:
Received 3 July 2015
Received in revised form
7 December 2015
Accepted 5 January 2016

Keywords:
Weightless neural networks
Mental images
Memory transfer

ABSTRACT

DRASiW is an extension of the WiSARD Weightless Neural Network (WNN) model with the capability of storing the frequencies of seen patterns during the training phase in an internal data structure called "mental image" (MI). Due to this capability, in a previous work it was demonstrated how to reversely process MIs in order to generate synthetic prototypes. Then, a training set composed of synthetic prototypes can be used to train new DRASiW systems (clones) with different architectures. In this paper we present a methodology to transfer memory between DRASiW systems, and we show how it is possible to generate clones of DRASiW systems with good classification capabilities within an acceptable loss of accuracy.

1. Introduction

Artificial Neural Networks (ANN) are a powerful and general technique for Machine Learning (ML). Unfortunately, one of the most significative short-coming is that a trained ANN is essentially a sort of *black box* [1]. Two main methodologies have been introduced to take advantage of the knowledge acquired by trained ANN: Transfer Learning (TL) and Rule Extraction (RE). These different methodologies of transferring, and then to take advantage of, the previously acquired knowledge on a source problem by an ANN can be both dated back to the early 1990s [2,3]. One has to point out that the aims of these two methodologies are different. In TL, the knowledge from Artificial Neural Networks (ANN) that have learned one task can be reused on related tasks in a process that is called "transfer" [2]. In such cases, *knowledge transfer* or *transfer learning* between task domains would be desirable. In particular, *transfer learning* aims to extract the knowledge from one or more *source tasks* and to apply this knowledge to a *target task* [4]. The most important distinction between different types of transfer learning in ANN is *representational* versus *functional*. The former is based on the idea to literally copy the trained network and train it on the new task [5]; while, the latter does not involve the explicit assignment of prior task representation to a new task, rather it employs the opportunity of taking advantage of supplemental training examples [6]. As reported in [2], source data is typically part of the training process in functional transfer whenever the source training data are available. Different theoretical approaches [7–9] and applications have been summarized in two recent surveys [10,11]. What is worth noticing is that the best results of the use of TL methodology nowadays are those reached in image and video processing [12–14]. In fact, as demonstrated by Yosinki, et al. [15] TL from deep convolutional network trained on large amounts of data has been shown to be effective across image datasets: image features extracted by convolutional networks which have been exposed to large image datasets generalize to many other domains [16].

Conversely, the underlying idea in RE is to use trained ANN as a vehicle for synthesizing the knowledge that is crucial for the success of knowledge-based systems [17]. In fact, the RE algorithms generally extract the hidden knowledge in terms of production rules. In this way, the problem to be solved remains the same, but the system to face it is now different. This is typical of domains in which we do not have any experience and knowledge on how to solve problems in it: we have just a certain amount of data. The most interesting aspect of RE is that we can discover interesting relationships existing within the data given and what general conclusions can be drawn. The number of new applications and theoretical contributions makes RE still a very active and interesting research field [18–20]. Contribution to RE has been given even with WNN [21,22].

In this paper we propose a different approach to transfer learning in WNN, in particular in the DRASiW [23] neural model of computation.[1] The new process, called *memory transfer*, is neither based on the availability of source training data nor on the aim of approaching

* Corresponding author.
E-mail addresses: massimo.degregorio@cnr.it (M. De Gregorio), maurizio.giordano@cnr.it (M. Giordano).

[1] Refer to Section 2.2 for the meaning associated to the word DRASiW.

http://dx.doi.org/10.1016/j.neucom.2016.01.087
0925-2312/

related tasks. In particular, we focus the attention on how the functionalities of a DRASiW system can be transferred to another DRASiW system but with a different architecture. The aim of this work is to use the knowledge of the source system in order to create clones with different architectures but with the same functionalities.

As far as we know, this is the very first approach of cloning WNN systems. The cloning process is carried out by extracting information from the neural internal status of the source network and by using this information to reproduce in one shot a neural internal status in the target network system with a different architecture, all this preserving the functionalities in the same domain within a tolerable error. The proposed DRASiW cloning mechanism is based on a particular characteristic these systems have: the learned knowledge about a domain is internally represented by a machine-readable data structure of DRASiW, called "mental image". In a previous work [24] we demonstrated how to reversely process mental images of DRASiW systems in order to generate synthetic prototypes. In this work we propose a methodology to transfer memory between DRASiW systems, and we show how it is possible to generate clones of DRASiW systems with good classification capabilities within an acceptable loss of accuracy. The cloning of knowledge is based on information extracted from mental images of the source DRASiW system.

A different approach of processing mental images to extract and make explicit the internal learnt knowledge of a DRASiW system is reported in [21]. Here the authors do not aim to transfer learning between ANNs, but rather to extract information from DRASiW mental images to generate production rules. The resulting rule-based classifier is comparable to decision tree classifiers in terms of performance.

In Section 2, we are going to introduce the two weightless neural models involved in this work: WiSARD and DRASiW. In Section 3, we will describe how to transfer classification and recognition capabilities between DRASiW systems through their mental images. In Section 4, comparison and discussion of experimental results of DRASiW original vs clone system on ten datasets are reported. In Section 5, concluding remarks and some future work perspectives are reported.

2. Weightless neural systems

Weightless Neural Networks (WNNs) [25,26], differently from classical Artificial Neural Networks, adopt a RAM-based model of neuron by which learned knowledge about a data domain is stored into RAM nodes instead of computed weights of neuron connections. WNNs have a basis for their biological plausibility because of the straightforward analogy between the address decoding in RAMs and the integration of excitatory and inhibitory signaling performed by the neuron dendritic tree. A RAM-based neuron receives an n-bit input (n-tuple) that is interpreted as a unique address (*stimulus*) of a RAM memory cell, used to access it either in writing (*learning*) or reading (*classification*) mode.

In the following two subsections we are going to introduce the two WNN models involved in our discussion.

2.1. The WiSARD model

While the use of n-tuple RAM nodes in pattern recognition problems is old [27], dating about 60 years, with the availability of integrated circuit memories in the late 1970s, the WiSARD (*W*ilkes, *S*tonham and *A*leksander *R*ecognition *D*evice) was the first artificial neural network machine to be patented and produced commercially [28]. In fact, WiSARD can be developed directly on reprogrammable hardware providing fast and flexible learning algorithms.

A WiSARD is composed of a set of classifiers, called *discriminators*, each one assigned to learn binary patterns belonging to a particular category/class. Therefore, a WiSARD has as many discriminators as the number of categories/classes it should be able to distinguish. The WiSARD is also called "multi-discriminator architecture".

Each discriminator is formed by a set of RAM nodes which store the information of occurrences of binary patterns during the learning phase. Given a binary pattern of size s, the so-called *retina*, it can be classified by a set of WiSARD discriminators, each one having x RAMs with 2^n memory cells such that $s = x \times n$. Since each RAM cell is uniquely addressed by an n-tuple of bits, the input pattern can be partitioned into a set of n-tuples, each one addressing one memory cell in exactly one RAM. n-tuples are pseudo-randomly selected and biunivocally mapped to RAMs (see the right part of Fig. 1), in such a way that the retina is completely covered.

Thus, in the general case, the random mapping statically extracts from the binary input pattern (stored in the retina) a number x of n-tuples of bits, and each n-tuple represents a binary address with n digits. This address is used to stimulate a RAM neuron (by accessing one of its cells) either in writing mode (learning phase) or reading mode (classification phase). Nevertheless, since any kind of information can be coded in binary patterns, by means of ad hoc data transformations, a WiSARD discriminator is able to learn then recognize data in both symbolic and numeric domains.

In order to train the discriminators, one has to set to 0 the content of all RAMs' cells (initialization), and choose a training set (TS) formed by binary patterns of $x \times n$ bits. For each pattern in TS, a 1 is stored in the memory cell of each RAM addressed by the training pattern. The information stored in RAM nodes during the training phase is used to deal with previously unseen patterns.

During the classification phase, RAM cells addressed by the input pattern are read and summed by the adder Σ and divided by x (number of RAM nodes). The number r thus obtained, called the *discriminator response*, is a sort of "similarity measure" of the input pattern with respect to the patterns in TS.

It is easy to see that $r=1$ if the input pattern belongs to the training set, $r=0$ if none of its n-tuples occurs in the training set. The closer is r to 1 the more "similar" is the input pattern to those patterns in the training set. The adder operator Σ enables this network of RAM nodes to exhibit – just like other ANN models based on synaptic weights – generalization and noise tolerance [29].

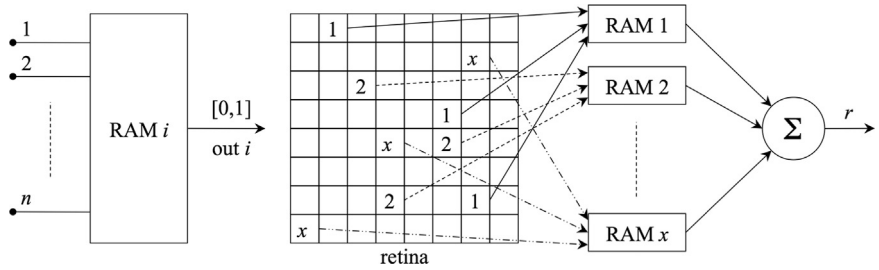

Fig. 1. RAM-neuron (left) and a WiSARD discriminator (right).

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

2.2. The DRASiW model

DRASiW [23] is an extension of the WiSARD model. During DRASiW training phase, memory cells accessed in write mode are incremented by 1 instead of being set to 1. Thus, at the end of the training phase, RAM cells store the number of occurrences (frequency) of a specific n-tuple of bits across training samples. In reading mode, the adder Σ counts the number of non-zero values stored in the addressed RAM cells. Therefore, the new domain of memory cells' contents (non-negative integers) produces the same classification capability of a WiSARD.

The DRASiW enhances the WiSARD with a backward-classification capability [30,31], i.e., each discriminator is able to produce representative examples (*prototypes*) of a class that have been learnt from trained patterns. In order to make this possible, RAM cells act as access counters, whose contents can be reversed to an internal "retina", where a "mental image" is produced, thus yielding a bidirectional structure. More specifically, this new form of bidirectional behavior allows DRASiW systems to provide, given the class name C in input, a gray-level, non-crisp example of C (as an example, see Fig. 2). Furthermore, this is why the system name is DRASiW: it recalls the name WiSARD spelled backwards to point out the new form of bidirectional behavior of DRASiW.

The "mental image" metaphor, associated with the internal "retina" metaphor, was originally explored in [32] in which the authors discuss the cognitive plausibilities related to these ideas.

3. From mental images to synthetic training set

Let us consider the mental image on the right side of Fig. 2. The image gray levels represent how many times the pixels were present in the instances of the original training set (some of these instances are shown on the left side of Fig. 2). The darker is the pixel, more frequent is the bit associated to it. This is the only knowledge a mental image carries with it. In fact, we have no information about how the subpatterns (groups of correlated pixels) form the RAM addresses in each training sample. Hence, we expect that a memory transfer mechanism based on mental images will be characterized by a degradation in system performance although it might preserve the mental image.

The mental images can be depicted in 3D where the new dimension is represented by the pixel gray levels. Fig. 3 shows in 3D (right side) the mental image of class "2" (left side).

What we get is a sort of city downtown upper view where pixels are represented by skyscrapers. All the information is stored in the downtown except for the way it was built: we see the result, but we do not know the process that built it.

From the 3D representation of the mental image, we can create a *Synthetic Training Set* (STS) by considering the meaning of gray levels, that is how many times those pixels were black in the images forming the original TS. The highest gray level will represent the possible number of events, all the other gray levels will represent distinct favorable events.

In order to reproduce a synthetic and plausible TS, we randomly distribute the building floors along the size of the highest one. So doing, we generate a new downtown formed by skyscrapers having all the same height but with real floors and missing floors from the ground to the top. At this point we start sinking the downtown floor by floor. At every sinking step, the patterns of the building floors are going to form the instances (prototypes) of the new STS. In Fig. 4, prototypes of class "2" generated by the mental image of Fig. 3 are reported. The STS is now used to train a new DRASiW system: the *clone* system. More details about how the prototypes generation algorithm works are reported in Section 3.2.

The original system is mainly characterized by (a) the number of classes; (b) the retina size; (c) the n-bit addressing of RAMs. The number of classes fixes the number of DRASiW discriminators; the retina size together with the n-bit addressing determines the number of neurons for each DRASiW discriminator. The clone and the original system share only the number of discriminators and the capability of generating the same mental images. By assuming the same retina, both the n-bit addressing and, consequently, the number of neurons could be different between the clone and the original system.

3.1. Data transformation

Although DRASiW was originally conceived as a pattern recognition device mainly focusing on image processing domain, with ad hoc data transformation, DRASiW can also be used successfully as multiclass classifier in the machine learning domain.

Indeed, if we consider numeric data domains in which each datum can be represented by a vector of features (attributes), we can adopt the well-known LibSVM [33] format to represent numeric data such that each datum (sample) of a training set can be represented in the form:

$$sample = \langle c_i, f_0 : v_0, \ldots, f_j : v_j \rangle; \tag{1}$$

where c_i is the identifier (a string or a number) of the class the datum belongs to, f_j and v_j are respectively the feature identifier (a string or a number), and its value (a real number) inside the feature vector representing the datum.

Before the data could be processed by a DRASiW multi-discriminator system, they have to be converted to binary patterns. First of all, feature values v in the range $[\alpha, \beta]$ have to be discretized and scaled to an integer value $\underline{v}$ in the interval $[0, t]$. With this transformation, any real number $v \in [\alpha, \beta]$ will be represented by the non-negative integer:

$$\underline{v} = \left\lceil \frac{(v-\alpha) \times t}{\beta - \alpha} \right\rceil. \tag{2}$$

Thus, under the transformation of Eq. (2), the format of dataset samples becomes

$$sample = \langle c_i, f_0 : \underline{v}_0, \ldots, f_j : \underline{v}_j \rangle; \tag{3}$$

where the feature vector has as elements all non-negative integers in the range $[0, t]$. For example, let us consider a training set of class c, called TS_c, composed of 4 samples ($|TS_c| = 4$). Suppose that, after

Fig. 2. Partial TS for class "2" and its corresponding mental image.

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

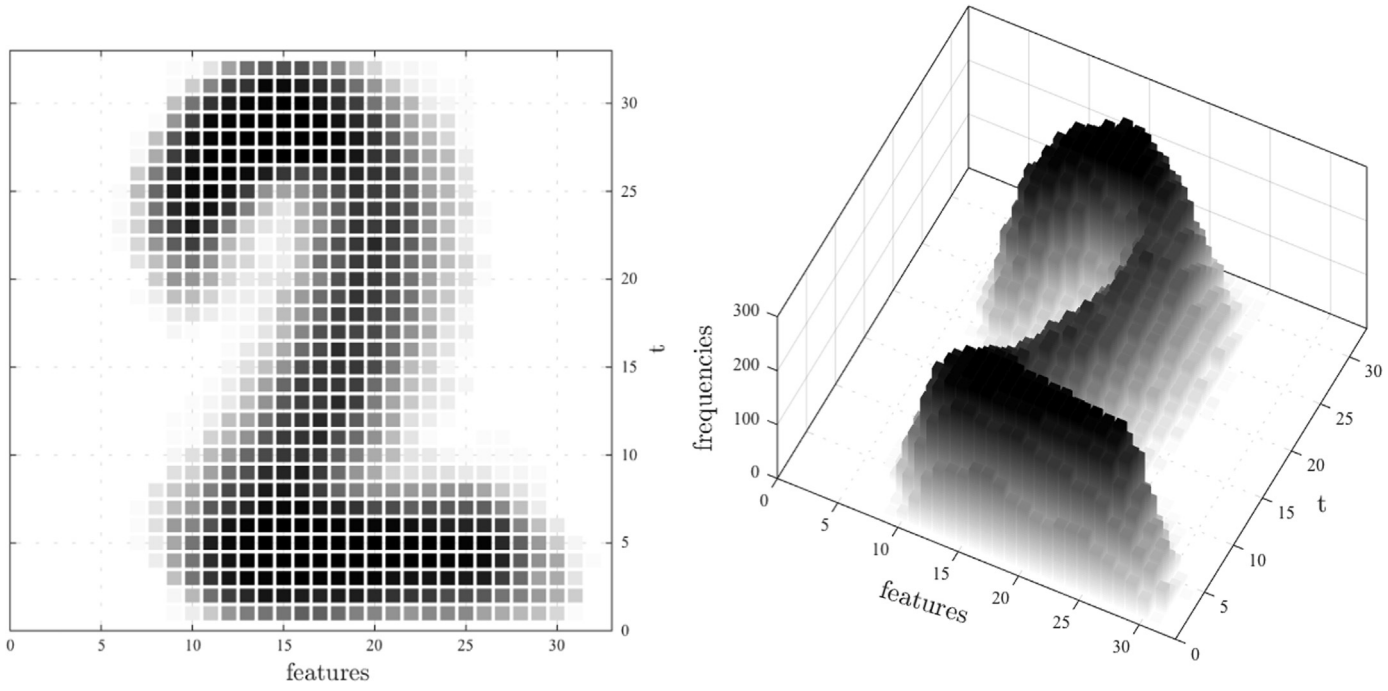

Fig. 3. Mental image and its 3D representation.

Fig. 4. Part of the original training set (top) and synthetic training set (bottom).

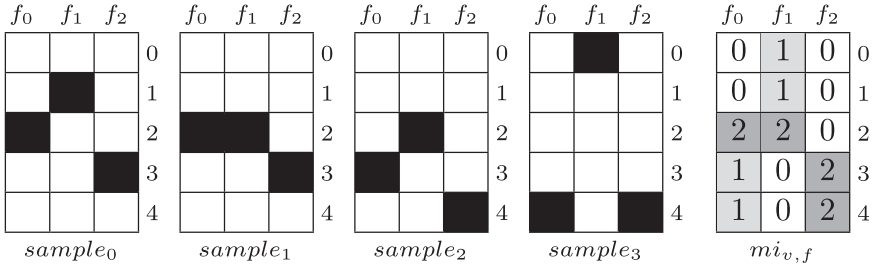

Fig. 5. Binary patterns representing samples of Eq. (4) and the corresponding mental image (right side).

scaling and discretization of feature values to integers in the range $[0,4]$ (t=4), TS_c will contains the following samples:

$$sample_0 = \langle 1, f_0 : 2, f_1 : 1, f_2 : 3\rangle$$
$$sample_1 = \langle 1, f_0 : 2, f_1 : 2, f_2 : 3\rangle$$
$$sample_2 = \langle 1, f_0 : 3, f_1 : 2, f_2 : 4\rangle$$
$$sample_3 = \langle 1, f_0 : 4, f_1 : 0, f_2 : 4\rangle. \quad (4)$$

Hence, the samples can be represented by the binary patterns of Fig. 5 whose corresponding mental image is sketched on the right side of the figure.

With the transformation of Eq. (2), numeric datasets could be used both for training and for classification in DRASiW systems. With other ad hoc transformations, DRASiW can even deal with non-numeric data, like nominal[2] and ordinal datatypes in symbolic domains [34].

3.2. Prototypes generation

Let us assume that the DRASiW classifier is composed of d discriminators, each one able to classify test samples in one of the classes $c_1, \ldots, c_d$, and trained on the corresponding $TS_{c_1}, \ldots, TS_{c_d}$. We assume the training sets, originally available in LibSVM format, have been transformed in the form of Eq. (3) before they are used for training.

[2] Also called *categorical*.

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

$f = f_0$	$V \leftarrow \{3,2,0,1,4\}$	$D \leftarrow \{0,3,1,2\}$
	$v \leftarrow v_3 = 3$	$\Rightarrow mi_{v_3,f_0} = 1(\downarrow 0) \Rightarrow x \leftarrow 0 \Rightarrow proto_0 \leftarrow \langle 1,\ldots,\ f_0 : v_3,\ldots\rangle$
	.	$\Rightarrow mi_{v_3,f_0} = 0$ **(end while)**
	$v \leftarrow v_2 = 2$	$\Rightarrow mi_{v_2,f_0} = 2(\downarrow 1) \Rightarrow x \leftarrow 3 \Rightarrow proto_3 \leftarrow \langle 1,\ldots,\ f_0 : v_2,\ldots\rangle$
	.	$\Rightarrow mi_{v_2,f_0} = 1(\downarrow 0) \Rightarrow x \leftarrow 1 \Rightarrow proto_1 \leftarrow \langle 1,\ldots,\ f_0 : v_2,\ldots\rangle$
	.	$\Rightarrow mi_{v_2,f_0} = 0$ **(end while)**
	$v \leftarrow v_0 = 0$	$\Rightarrow mi_{v_0,f_0} = 0$
	$v \leftarrow v_1 = 1$	$\Rightarrow mi_{v_1,f_0} = 0$
	$v \leftarrow v_4 = 4$	$\Rightarrow mi_{v_4,f_0} = 1(\downarrow 0) \Rightarrow x \leftarrow 2 \Rightarrow proto_2 \leftarrow \langle 1,\ldots,\ f_0 : v_4,\ldots\rangle$
	.	$\Rightarrow mi_{v_4,f_0} = 0$ **(end while)**
$f = f_1$	$V \leftarrow \{0,2,1,3,4\},$	$D \leftarrow \{2,3,0,1\}$
	$v \leftarrow v_0 = 0$	$\Rightarrow mi_{v_0,f_1} = 1(\downarrow 0) \Rightarrow x = 2 \Rightarrow proto_2 \leftarrow \langle 1,\ldots,\ f_1 : v_0,\ldots\rangle$
	.	$\Rightarrow mi_{v_0,f_1} = 0$ **(end while)**
	$v \leftarrow v_2 = 2$	$\Rightarrow mi_{v_2,f_1} = 2(\downarrow 1) \Rightarrow x = 3 \Rightarrow proto_3 \leftarrow \langle 1,\ldots,\ f_1 : v_2,\ldots\rangle$
	.	$\Rightarrow mi_{v_2,f_1} = 1(\downarrow 0) \Rightarrow x = 0 \Rightarrow proto_0 \leftarrow \langle 1,\ldots,\ f_1 : v_2,\ldots\rangle$
	.	$\Rightarrow mi_{v_2,f_1} = 0$ **(end while)**
	$v \leftarrow v_1 = 1$	$\Rightarrow mi_{v_1,f_1} = 1(\downarrow 0) \Rightarrow x = 1 \Rightarrow proto_1 \leftarrow \langle 1,\ldots,\ f_1 : v_1,\ldots\rangle$
	.	$\Rightarrow mi_{v_1,f_1} = 0$ **(end while)**
	$v \leftarrow v_4 = 4$	$\Rightarrow mi_{v_4,f_1} = 0$
$f = f_2$	$V \leftarrow \{1,0,4,2,3\}$	$D \leftarrow \{2,3,0,1\}$
	$v \leftarrow v_1 = 1$	$\Rightarrow mi_{v_1,f_2} = 0$
	$v \leftarrow v_0 = 0$	$\Rightarrow mi_{v_0,f_2} = 0$
	$v \leftarrow v_4 = 4$	$\Rightarrow mi_{v_4,f_2} = 2(\downarrow 1) \Rightarrow x = 2 \Rightarrow proto_2 \leftarrow \langle 1,\ldots,\ f_2 : v_4,\ldots\rangle$
	.	$\Rightarrow mi_{v_4,f_2} = 1(\downarrow 0) \Rightarrow x = 3 \Rightarrow proto_3 \leftarrow \langle 1,\ldots,\ f_2 : v_4,\ldots\rangle$
	.	$\Rightarrow mi_{v_4,f_2} = 0$ **(end while)**
	$v \leftarrow v_2 = 2$	$\Rightarrow mi_{v_2,f_2} = 0$
	$v \leftarrow v_3 = 3$	$\Rightarrow mi_{v_3,f_2} = 2(\downarrow 1) \Rightarrow x = 0 \Rightarrow proto_0 \leftarrow \langle 1,\ldots,\ f_2 : v_3,\ldots\rangle$
	.	$\Rightarrow mi_{v_3,f_2} = 1(\downarrow 0) \Rightarrow x = 1 \Rightarrow proto_1 \leftarrow \langle 1,\ldots,\ f_2 : v_3,\ldots\rangle$
	.	$\Rightarrow mi_{v_3,f_2} = 0$ **(end while)**

Fig. 6. Trace of Algorithm 1 on the mental image of Fig. 5 given as input.

The synthetic prototypes' generation procedure is described in Algorithm 1. Given a class c, for each feature identifier f the procedure assigns a feature value v in the interval $[0,t]$ to as many prototypes as the number of occurrences stored in the corresponding element $m_{v,f}$ of the mental image.

Algorithm 1. Prototypes generation.

```
base ← 0
for c ∈ {c_1, ..., c_d} do
  for f ∈ {f_0, ..., f_j} do
    V ← RndPerm({0, ..., t})
    D ← RndPerm({0, ..., |TS_c|})
    for v ∈ V do
      while mi_{v,f} > 0 do
        d ← extractFrom(D)
        mi_{v,f} ← mi_{v,f} − 1
        proto_{base+d} ← ⟨c, ..., f : v, ...⟩
      end while
    end for
  end for
  base ← base + |TS_c|
end for
```

As an example of prototypes production starting from a mental image of a DRASiW system, let us consider as input to Algorithm 1 the mental image depicted in the right side of Fig. 5. This mental image is the result of training a system with the samples of Eq. (4). By executing Algorithm 1 on this mental image, we obtain the execution trace showed in Fig. 6, that generates the following synthetic dataset of prototypes:

$$\begin{aligned} proto_0 &= \langle 1, f_0 : 3, f_1 : 2, f_2 : 3\rangle \\ proto_1 &= \langle 1, f_0 : 2, f_1 : 1, f_2 : 3\rangle \\ proto_2 &= \langle 1, f_0 : 4, f_1 : 0, f_2 : 4\rangle \\ proto_3 &= \langle 1, f_0 : 2, f_1 : 2, f_2 : 4\rangle. \end{aligned} \tag{5}$$

As the reader can notice, generated prototypes are quite similar to the input samples. In fact, $proto_1$ and $proto_2$ are respectively equal to $sample_0$ and $sample_3$, while the others differ (see Fig. 7).

4. Experimental results

To evaluate the efficiency of the cloning process we compare classification accuracy measurements of the original and clone DRASiW system. The comparison was carried out on ten different datasets selected from the following repositories: KEEL Repository,[3] Machine Learning Data Set Repository,[4] and UCI

[3] http://sci2s.ugr.es/keel
[4] http://mldata.org

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

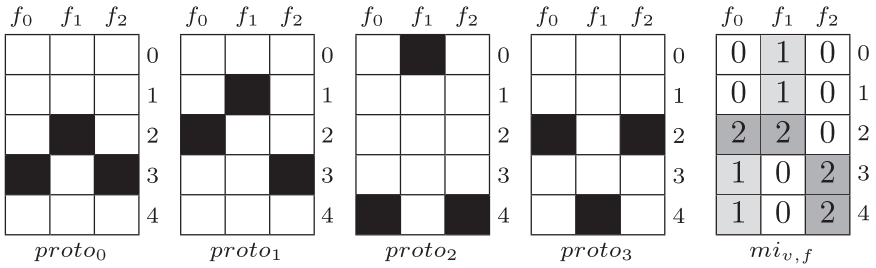

Fig. 7. Binary patterns representing prototypes of Eq. (5) and the corresponding mental image (right side).

Table 1
List of datasets used in experiments, their domain, feature types (R=real, I=integer, N=nominal), size, number of classes, and source.

Dataset	Description	Features no. (R/I/N)	Train/test sizes	Class no.	Source
crx	Predict whether a subject has diabetes	15 (3/3/9)	587/66	2	Keel
ecoli	Predict localization site of proteins by cell measures	7 (7/0/0)	302/34	8	Keel
har	Human activity recognition using smartphone	561 (R/0/0)	7352/2947	6	UCI
ionosphere	Determine if a given radar signal shows evidence of free electrons	33 (32/1/0)	315/36	2	Keel
optdigits	Optical recognition of handwritten digits	64 (0/64/0)	5058/562	10	Keel
pima	Predict whether a subject has diabetes	8 (8/0/0)	691/71	2	Keel
satimage	Landsat satellite application	36 (0/36/0)	5791/644	6	Keel
segment	Image segmentation application	19 (19/0/0)	2079/231	7	Keel
splice	Splice-junction gene sequences recognition	60 (0/0/60)	2857/318	3	Keel
svmguide1	Astroparticle application	4 (R/0/0)	3089/4000	2	MLData

Machine Learning Repository.[5] Datasets were chosen so as to be very different between them with respect to the problem:

1. domain (i.e., physics, social, medical, etc.);
2. dimension (i.e., number of features and number of samples);
3. datatype (i.e., integer, real, nominal).

The list of datasets with their sources and dimensions is reported in Table 1.

Before starting our experimental study of the DRASiW transfer mechanism proposed in this work, we first provide some insights and a comparison of classification capabilities of DRASiW with respect to other state-of-the-art machine learning methods, when all methods accomplish training and classification by cross-validation on the set of considered datasets (see Table 1).

A machine learning method based on DRASiW model has been implemented as additional package of the sklearn machine learning library.[6] As such, the developed DRASiW-based machine learning method is compliant to its programming interface. We measured classification accuracy of all methods on the chosen datasets, by averaging the accuracies obtained in 50 repetitions of a ten-fold cross validation on each dataset. Random splits of each dataset were prepared to form 50 pairs of train and test sets. All methods worked on the same 50 pairs of sample sets. In all these preliminary experiments no features' selection was carried out, as well as the default configuration was used for all methods independently on the target problem domain, size and feature type combination. The DRASiW classifier's default configuration was set to 16-bit ($n=16$) addressing scheme and to 1024-tics scaling ($t=1024$). The average accuracies of all methods are reported in Table 2, where underlined (italic) numbers are the best (worst) results on the corresponding dataset. As we can notice, in the average the ensemble method GTB won over the others, although DRASiW showed comparable accuracies similar to GTB in most of the datasets.

From this point on, the focus of our experimental study is the performance comparison of DRASiW systems with their clones on the selected datasets. For each dataset we first generated the synthetic training set by running DRASiW at the largest scale, that is with $t=1024$. Then we carried out measures of original and clone system by varying both the retina size, i.e. the number t of scale tics used, and the n-bit addressing scheme of the system (from 2 to 64 bits). More in detail, in each experiment done with configuration (n,t), we trained the original DRASiW on the training set, while its clone was trained on the synthetic training set; then, we ran the two so-trained systems on the same test set for classification. In each experiment we measured the accuracies given by the following formulas:

$$a_i^{o(n,t)} = \frac{TP_i^{o(n,t)}}{|\,Testset_i\,|}, \quad a_i^{c(n,t)} = \frac{TP_i^{c(n,t)}}{|\,Testset_i\,|} \tag{6}$$

where $TP_i^{o(n,t)}$ ($TP_i^{c(n,t)}$) are the number of true positives (i.e. correct classifications of testing samples) provided by the original (clone) DRASiW in the (n,t) configuration on the i-th dataset.

Accuracies results of the experiments are reported in the plots of Figs. 8 and 9.[7] Depending on the dataset, we experienced that for some datasets the performance of the original and clone system is quite similar (see *crx*, *ecoli*, *pima* and *svmguide1*), while in other cases we measured an accuracy loss. This loss in the case of *har*, *satimage* and *segment* dataset, increases with n (bit-addressing scheme) with a maximum loss, respectively, ranging from 6% to 15%. In the case of *optdigits*, the clone shows an almost constant loss in accuracy with respect to the original system (around 5–6%) for all (n,t) configurations in which $n<64$. In the cases of 64-bit clone systems the accuracy decay is greater. Indeed, the random-generated correlation of data in each synthetic sample approximates the correlation of the original data worse as the number of neurons decreases.[8]

[5] http://http://archive.ics.uci.edu/ml
[6] http://scikit-learn.org

[7] For the lack of space we had to split the plots of ten experiments in two figures.
[8] Remind that the number of neurons is inversely proportional to n.

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

Table 2
DRASiW accuracy comparison with state-of-the-art classification methods: Multi-Layer Perceptron (MLP), Random Forrest (RF), Extremely Randomized Tree (ERT), Gradient Tree Boosting (GTB), Linear Discriminant Analysis (LDA), *k*-Nearest Neighbors (kNN), Logistic Regression (LR), and Support Vector Machine with RBF kernel (SVC).

Dataset	Method								
	DRASiW	MLP	RF	ERT	GTB	LDA	kNN	LR	SVC
crx	0.8500	0.8697	0.8582	0.8491	0.8706	0.8658	0.6576	0.8724	*0.5482*
ecoli	0.8547	0.7100	0.8241	0.8353	0.8118	0.7771	0.8294	0.7994	*0.4318*
har	0.9281	0.9505	0.9053	0.9152	0.9365	0.9623	*0.9016*	0.9620	0.9403
ionosphere	0.9289	0.8583	0.9328	0.9378	0.9456	0.8711	*0.8522*	0.8789	0.9378
optdigits	0.9841	0.9664	0.9648	0.9679	0.9783	0.9529	0.9883	0.9649	*0.7202*
pima	0.7639	0.7543	0.7338	0.7574	0.7574	0.7247	0.7779	*0.6421*	*0.6421*
satimage	0.8986	0.8349	0.9024	0.9030	0.9038	0.8413	0.9085	0.8308	*0.2654*
segment	0.9763	0.8833	0.9758	0.9758	0.9826	0.9167	0.9444	0.9284	*0.6272*
splice	0.9409	0.8423	0.9398	0.9334	0.9712	0.8452	*0.7816*	0.8484	0.9152
svmguide1	0.9623	0.9283	0.9690	0.9663	0.9705	0.8900	0.9660	0.9545	*0.6693*

It is more difficult to try explaining the performance decays in the case of *splice*. It is evident that the 2-bit addressing scheme of the clone is not efficient in reproducing the original data correlation. With other addressing schemes, the clone system accuracy follows the same trends of the original system accuracy, although with lower performance.

On the other side, the clone systems showed higher performances on *ionosphere* and *pima* dataset in certain (n,t) configurations. This improvement depends on two factors: (1) in such configurations the clone system efficiently reproduces the original data correlations; (2) since the transfer memory procedure soon generates the most significant samples, as well as the STS cardinality is lower than the TS cardinality, it is likely that, for the *ionosphere* and *pima* dataset, several non-significant samples are not inserted in the STS.

In order to further analyze the accuracy loss of clone systems we carried out a second experimentation only on the *satimage* dataset. We chose this dataset since it showed in the previous experiments a significant accuracy loss of the clone systems throughout all system configurations. Our intention was to verify whether having fewer samples in a training set may affect the accuracy loss. To this aim, in the new experiments we trained the systems on subsets (with different sizes) of the original dataset. More in detail, from the original dataset with 6435 samples, we randomly extracted a test set with 315 samples. The remaining 6120 samples were split into 5 distinct data chunks. Then, we prepared five experiments by constructing five training sets as follows: in the first experiment we used one data chunk for training, namely *satimage1*; in the second experiment we used for training the *satimage1* dataset plus a new data chunk, namely *satimage2*, and so on, up to the fifth experiment working on a training set with 6120 samples, which is named *satimage5*. Before running each experiment, we trained a DRASiW system with configuration (64, 1024) to generate the relative synthetic training set. As before, in each experiment we measured the classification accuracies of the DRASiW system and its clone under different (n,t) configurations across the five datasets which, this time, comes from the same data source. In all experiments, the two systems classified samples taken from the same test set.

Accuracy results are reported in the plots of Fig. 10. As we can notice, the experiments do not show significant increases of the accuracy loss while resizing the training set of samples.

4.1. Paired-test statistical comparison over datasets

In order to compare the DRASiW original and clone system we looked at the statistical literature to find an appropriate comparison method. We chose to compare performances of the original DRASiW and of its clone by using two tests: the Paired *T*-test and the Wilcoxon Signed-Ranks test [35].

The Paired *T*-test is a non-parametric test that checks whether the average difference between pairs of observations is not significantly different from zero. In our case study, pairs of observations are accuracies of the original and clone system, for a fixed configuration (n, t), across the N datasets. Thus for the Paired *T*-test we make the following set of null-hypotheses:

$H^T_{(n,t)} =$ *the average difference of accuracies* $a_i^{o(n,t)}$ *and* $a_i^{c(n,t)}$ *is not significantly different from zero.*

where $a_i^{o(n,t)}$ and $a_i^{c(n,t)}$ are the accuracies of the original and the clone system on the i-th dataset. Let $d_i^{(n,t)} = a_i^{o(n,t)} - a_i^{c(n,t)}$ and $\overline{d}^{(n,t)} = \frac{1}{N}\sum_{i=1}^{N} d_i^{(n,t)}$ be, respectively, the difference in accuracy of the two systems on the i-th dataset and the mean of accuracy differences on all datasets. The T statistic is computed as

$$T = \frac{\overline{d}^{(n,t)}}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(d_i^{(n,t)} - \overline{d}^{(n,t)})^2}} \tag{7}$$

and it is distributed according to the Student distribution with $N-1$ degrees of freedom.

One problem with the Paired *T*-test is the assumption that, unless the sample size (i.e. the number of datasets N) is large enough (>30), the distribution of differences $d_i^{(n,t)}$ of the two compared random variables is distributed normally. In our case, datasets are very different from each other and we cannot assume the difference distribution is normal. On the other hand, since in our case study $N<30$, we know that Kolmogorov–Smirnov and similar tests for testing the normality of distributions have little power on small samples, that is they are unlikely to detect normality and warn against using the Paired *T*-test.

The Wilcoxon Signed-Ranks' test is a non-parametric test which checks whether the median difference between pairs of observations is not significantly different from zero. As before, the pairs of observations to compare are accuracies of the original and clone system, for a fixed configuration (n,t), across the N datasets. Let us make the following set of null-hypotheses for the Wilcoxon Signed-Ranks test:

$H^W_{(n,t)} =$ *the median difference of accuracies* $a^o_{i_{n,t}}$ *and* $a^c_{i_{n,t}}$ *is not significantly different from zero.*

The Wilcoxon Signed-Ranks rank the differences $d_i^{(n,t)}$ in accuracy of the two classifiers over N datasets, ignoring the sign and comparing the ranks for the positive and negative differences.

The differences $d_i^{(n,t)}$ are ranked according to their absolute values; average ranks are assigned in case of ties. Let $R_+^{(n,t)}$ be the sum of ranks for the datasets on which the original DRASiW system outperformed its clone, and $R_-^{(n,t)}$ the sum of ranks for the opposite. Ranks of $d_i^{(n,t)} = 0$ are split evenly among the sums; if

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

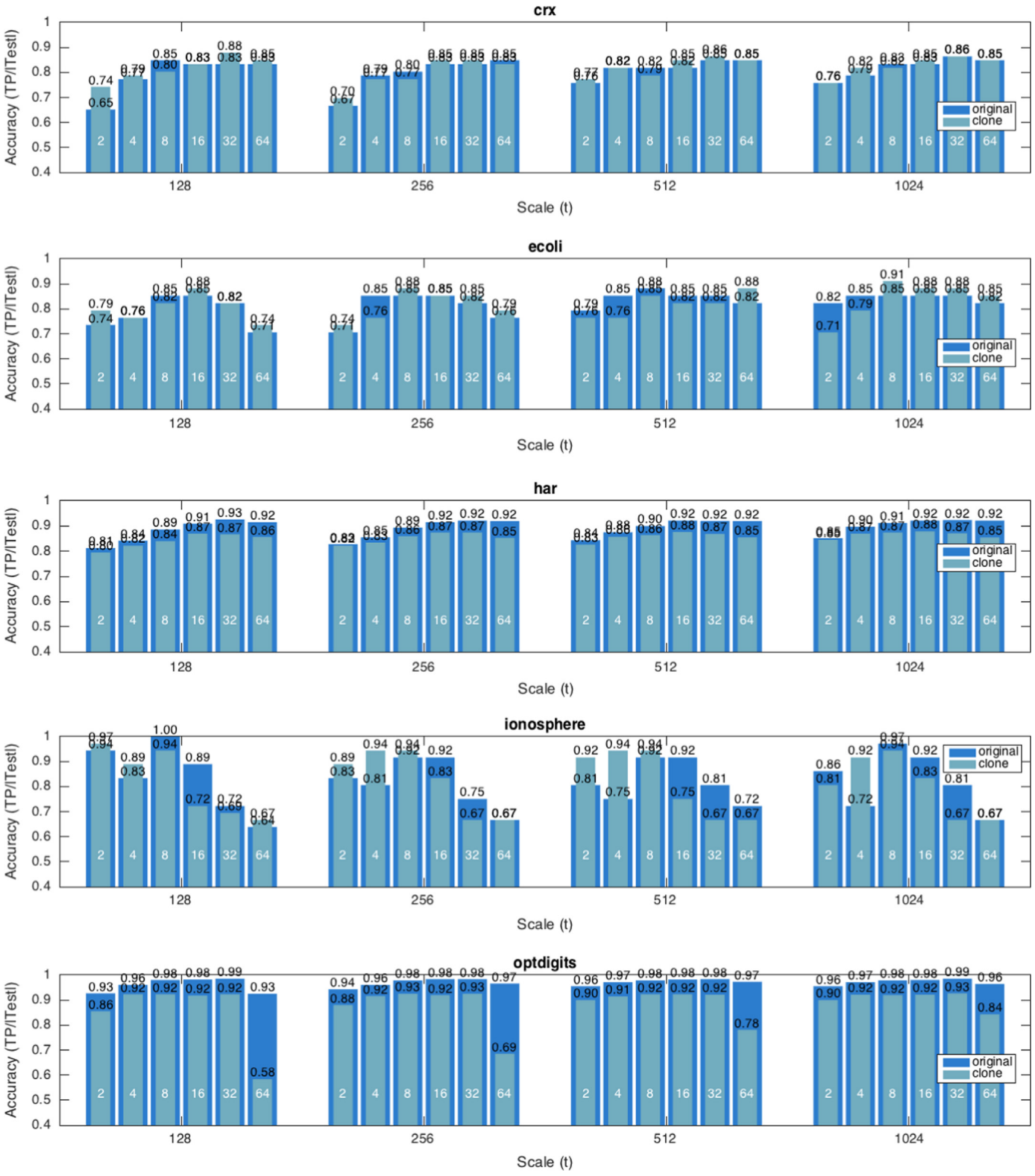

Fig. 8. Accuracy of the original and clone systems over 10 datasets (part 1).

there is an odd number of them, one is ignored:

$$R_+^{(n,t)} = \sum_{d_i^{(n,t)} > 0} rank(d_i^{(n,t)}) + \frac{1}{2} \sum_{d_i^{(n,t)} = 0} rank(d_i^{(n,t)})$$

$$R_-^{(n,t)} = \sum_{d_i^{(n,t)} < 0} rank(d_i^{(n,t)}) + \frac{1}{2} \sum_{d_i^{(n,t)} = 0} rank(d_i^{(n,t)}). \quad (8)$$

Let m be the smaller sum, $m = \min(R_+^{(n,t)}, R_-^{(n,t)})$. For a number $N \geq 25$ of datasets, the statistics

$$W = \frac{m - \frac{1}{4}N(N+1)}{\sqrt{\frac{1}{24}N(N+1)(2N+1)}} \quad (9)$$

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

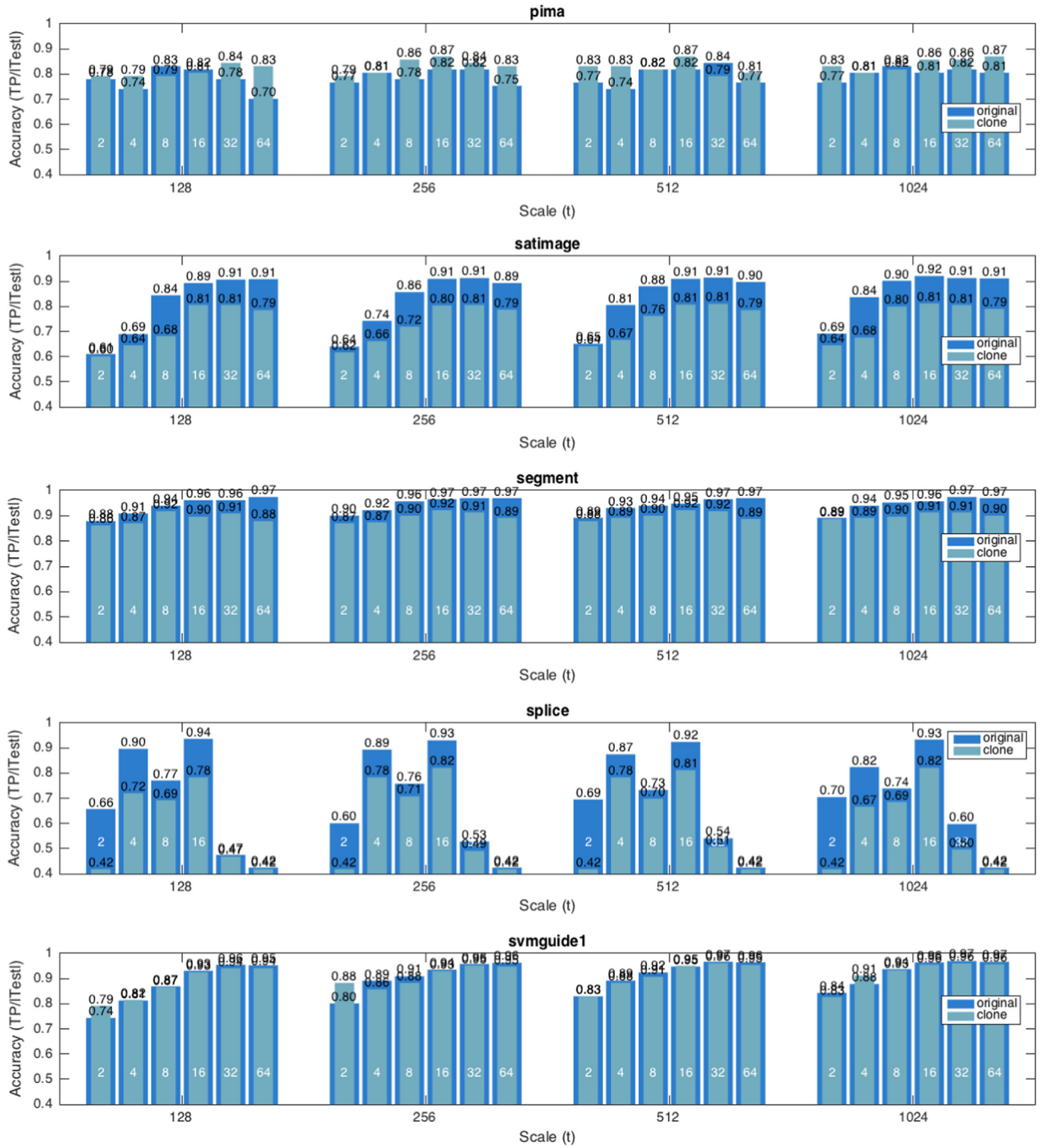

Fig. 9. Accuracy of the original and clone systems over 10 datasets (part 2).

is distributed approximately normally.[9]

In Table 3 the results of Wilcoxon Signed-Ranks test and the Paired T-test are reported in all possible configurations (n, t). We highlighted with gray color all null-hypotheses rejected by the two tests. With $\alpha = 0.05$, the null-hypotheses can be rejected if the corresponding p-value is smaller than α.

The Wilcoxon Signed-Ranks' test, in our case study, is more robust since it does not require the observation differences to have a normal distribution. For this reason, in the following discussion of the experimental results we will refer mainly to the Wilcoxon test.

The statistical analysis proves how the 8 and 32-bit addressing schemes causes, in most scaling configurations (128, 512, and 1024), a significant average difference of accuracies of the original and clone system, and the same holds in only one scaling configuration (128) for

[9] For $N < 25$ statistics books tables provides exact values of W.

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

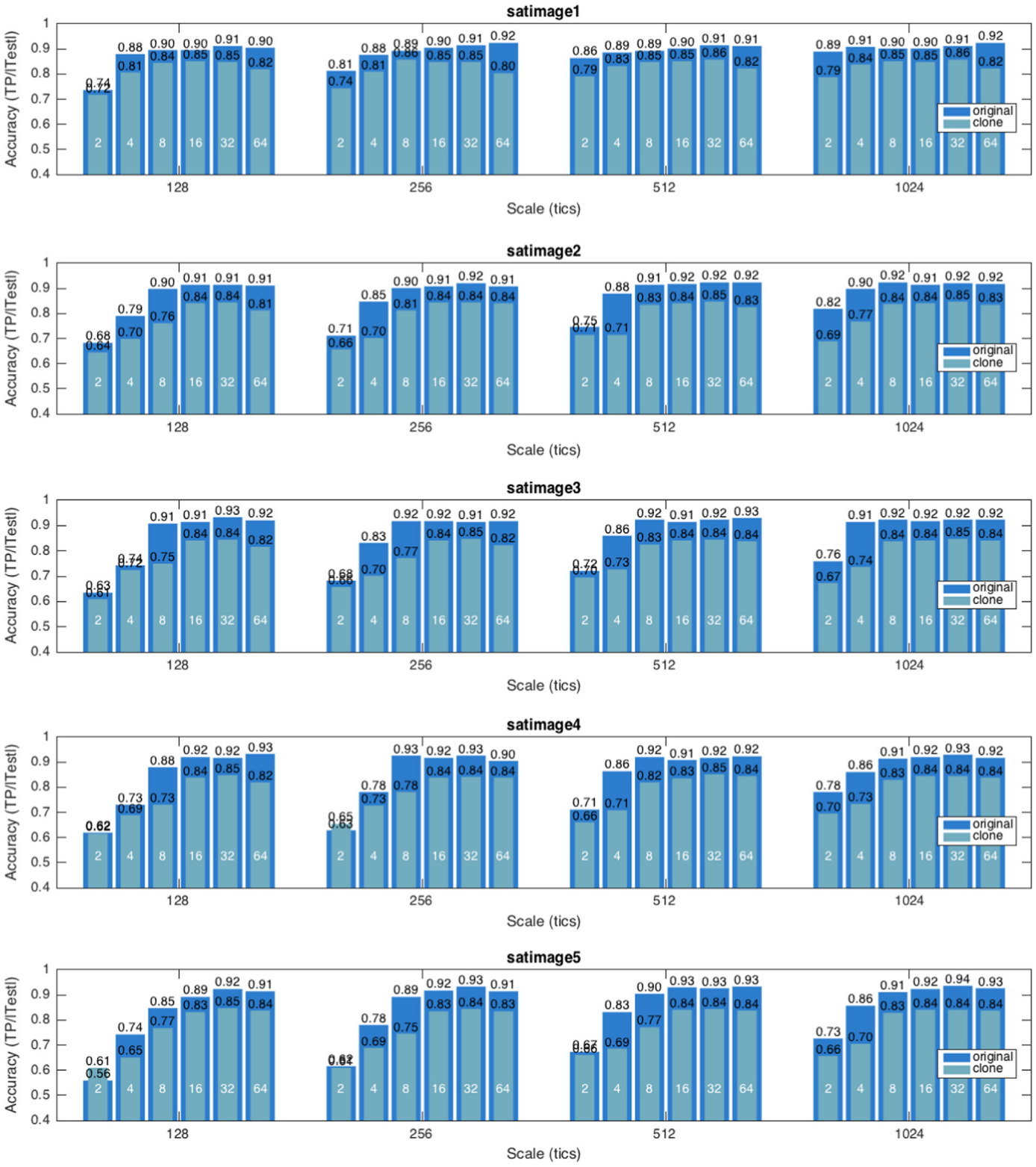

Fig. 10. Accuracy of the original and clone systems over subsampling of the *satimage* dataset.

16-bit addressing. This means that in each of those configurations, while summing the ranks of signed differences in accuracies of the two systems, the contributions in the positive signed differences (which correspond to better performances of the original system) are more relevant. This is also evident if we look at the bar gaps between the original and clone system of the plots in Figs. 8 and 9, by comparing these gaps along the dataset dimension, that is vertically and across graphs, Although the Wilcoxon test in our case study is more robust than the Paired *T*-test, it worth noticing how, after all, the results of the two tests agree.

In conclusion, by analyzing Table 3 we can assert that in the majority of the experimented (n,t) configurations and, by averaging measurements on the benchmarks datasets, the cloning process does not produce significant loss in accuracy. Of course,

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

Table 3
Paired *T*-test and Wilcoxon Signed-Ranks test results for original vs clone system comparison (statistics and *p*-values).

Hypothesis	T-Statistic	*p*-Value	Hypothesis	W-Statistic	*p*-Value
$H^T_{(2,128)}$	−0.4	0.7003	$H^W_{(2,128)}$	26.0	0.8785
$H^T_{(4,128)}$	−0.9	0.3771	$H^W_{(4,128)}$	18.0	0.5940
$H^T_{(8,128)}$	−3.9	0.0036	$H^W_{(8,128)}$	0.0	0.0051
$H^T_{(16,128)}$	−2.8	0.0224	$H^W_{(16,128)}$	3.0	0.0209
$H^T_{(32,128)}$	−1.3	0.2182	$H^W_{(32,128)}$	12.0	0.2135
$H^T_{(64,128)}$	−1.1	0.3065	$H^W_{(64,128)}$	21.0	0.5076
$H^T_{(2,256)}$	−0.3	0.7376	$H^W_{(2,256)}$	27.0	0.9594
$H^T_{(4,256)}$	−1.4	0.1989	$H^W_{(4,256)}$	9.0	0.1097
$H^T_{(8,256)}$	−1.4	0.2087	$H^W_{(8,256)}$	14.0	0.1688
$H^T_{(16,256)}$	−2.3	0.0490	$H^W_{(16,256)}$	7.0	0.0663
$H^T_{(32,256)}$	−2.2	0.0558	$H^W_{(32,256)}$	9.0	0.0593
$H^T_{(64,256)}$	−1.5	0.1723	$H^W_{(64,256)}$	10.5	0.1548
$H^T_{(2,512)}$	−0.7	0.5279	$H^W_{(2,512)}$	21.0	0.5076
$H^T_{(4,512)}$	−0.5	0.6212	$H^W_{(4,512)}$	15.0	0.3743
$H^T_{(8,512)}$	−2.8	0.0213	$H^W_{(8,512)}$	2.0	0.0152
$H^T_{(16,512)}$	−2.1	0.0608	$H^W_{(16,512)}$	11.0	0.0926
$H^T_{(32,512)}$	−3.6	0.0055	$H^W_{(32,512)}$	2.0	0.0093
$H^T_{(64,512)}$	−1.8	0.1059	$H^W_{(64,512)}$	8.0	0.0858
$H^T_{(2,1024)}$	−1.7	0.1153	$H^W_{(2,1024)}$	7.0	0.0663
$H^T_{(4,1024)}$	−0.8	0.4689	$H^W_{(4,1024)}$	14.0	0.3139
$H^T_{(8,1024)}$	−2.3	0.0471	$H^W_{(8,1024)}$	9.0	0.0593
$H^T_{(16,1024)}$	−2.0	0.0766	$H^W_{(16,1024)}$	11.0	0.0926
$H^T_{(32,1024)}$	−2.4	0.0379	$H^W_{(32,1024)}$	5.0	0.0382
$H^T_{(64,1024)}$	−1.5	0.1609	$H^W_{(64,1024)}$	7.0	0.1235

there are some (n,t) configurations in which this assertion does not hold. The main problem here is that, apart from exploring different options by experiments, at the moment we do not have a procedure to find, given a problem dataset, the optimal (n,t) configuration of the clone system, which is the receiver neural network of the knowledge transfer.

4.2. Marginal frequencies' statistical analysis on each dataset

In the previous section we have compared performances of the clone and the original system in the average over a set of ten datasets. In this section, for each fixed configuration (n,t) we examine more in detail the performance loss of the two systems for each dataset.

To this aim we decide to use the non-parametric McNemar test [36]. This test has been historically applied to statistically study the significant change of medical treatments. In general, the McNemar test verifies the occurring of differences in dichotomous data (presence/absence or positive/negative) of an event (e.g., the disease in a population), whenever data are available in the form of frequencies. The test aims to determine whether the marginal frequencies are the same.

Let us consider the original and clone system working in the same configuration (n,t) on the i-th dataset. In our case study we can consider the classification of the DRASiW in configuration (n,t) on the same samples before ($s^{o(n,t)}$) and after ($s^{c(n,t)}$) the cloning process has occurred.[10].

[10] Although up to now we referred to $s^{o(n,t)}$ and $s^{c(n,t)}$ has two different systems, like different entities, in such an interpretation we consider them as they were the same system in two temporal different states.

More in detail, the event frequencies under test are the positive/negative classifications of both systems on the i-th dataset, and the contingency table is the following:

	Positives of $s^{o(n,t)}$ on i-th dataset	Negatives of $s^{o(n,t)}$ on i-th dataset	
Positives of $s^{c(n,t)}$ on i-th dataset	a_i	b_i	$TP_i^{c(n,t)} = a_i + b_i$
Negatives of $s^{c(n,t)}$ on i-th dataset	c_i	d_i	$c_i + d_i$
Column total	$TP_i^{o(n,t)} = a_i + c_i$	$b_i + d_i$	

Let us define the following set of hypotheses:

$H^N_{(n,t),i}$ = *the marginal probabilities* P_{b_i} *and* P_{c_i} *are not significantly different for the case of the* $i-th$ *dataset*.

The McNemar Test statistic with the Edward continuity corrections [37] is

$$\chi^2 = \frac{(|b-c|-1)^2}{b+c} \tag{10}$$

and it has a χ^2 distribution with 1 degree of freedom. The null-hypotheses $H^N_{(n,t),i}$ are rejected whether p-value is less than $\alpha = 0.05$. In Table 4 we have reported for each dataset the p-values computed by the McNemar test, and with gray color we highlighted the rejected hypotheses. As one can notice, the datasets for which we get the majority of rejected hypotheses are the same for which, by looking at the plots of Figs. 8 and 9, we appreciate the larger accuracy gaps between the original and clone system (e.g., *har*, *optdigits*, *satimage*, *segment*, *splice*, and *svmguide1*).

The *ionosphere* case has the majority of accepted null-hypotheses. This result seems to be quite strange. Indeed, by looking (see Fig. 8) at the very frequent accuracy gaps between the clone and original system (in both positive and negative direction), we would have expected that most of the McNemar hypotheses had been rejected. The explanation for this strange behavior is due to the fact that the test may result less accurate with small sample size. In the *ionosphere* dataset we had 36 test samples: in the $(16, 256)$ configuration the original system provided 33 TPs ($33/36 \approx 0.92$ accuracy) and the clone gave 28 TPs ($28/36 \approx 0.78$ accuracy); hence, $a=27$, $b=1$, $c=6$ and $d=2$, with a $\chi^2 \approx 2.29$ and a p-value$=0.045$, which makes the hypotheses rejected. Conversely, in the $(4, 512)$ configuration the original system provided 28 TPs (0.75 accuracy) and the clone gave 34 TPs ($34/36 \approx 0.94$ accuracy); hence, $a=2$, $b=8$, $c=1$ and $d=1$, with a $\chi^2 \approx 4$ and a p-value$=0.131$, which makes the hypotheses accepted. As a conclusion, we expect that if the *ionosphere* dataset had had more testing samples, most of the rejected hypotheses would have been accepted.

4.3. Comparing by mental images

We also evaluated the systems in terms of the capability of generating plausible mental images: all the mental images generated by the clone systems are *exactly the same produced by the original systems*. This is due to the procedure that generates mental images in a DRASiW system: the mental image information does not depend on the original system architectures (number of neurons) and configurations (number of bits used in RAM addressing), but only on the discretization scale (t). In fact, the retina dimension is characterized

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

Table 4
McNemar test results for original vs clone system comparison (p-values).

Hypothesis	Dataset									
	crx	*ecoli*	*har*	*ionosphere*	*optdigits*	*pima*	*satimage*	*segment*	*splice*	*svmguide1*
$H^N_{(2,128)}$	0.15	0.48	0.00	1.00	0.00	1.00	0.50	0.29	0.00	0.00
$H^N_{(4,128)}$	1.00	0.72	0.00	0.62	0.00	0.22	0.00	0.03	0.00	0.56
$H^N_{(8,128)}$	0.37	1.00	0.00	0.48	0.00	0.50	0.00	0.27	0.00	0.51
$H^N_{(16,128)}$	0.62	1.00	0.00	0.08	0.00	1.00	0.00	0.00	0.00	0.13
$H^N_{(32,128)}$	0.25	0.68	0.00	1.00	0.00	0.18	0.00	0.01	0.87	0.00
$H^N_{(64,128)}$	1.00	1.00	0.00	1.00	0.00	0.03	0.00	0.00	0.48	0.00
$H^N_{(2,256)}$	0.75	1.00	0.29	0.62	0.00	0.72	0.07	0.05	0.00	0.00
$H^N_{(4,256)}$	1.00	0.25	0.00	0.07	0.00	0.68	0.00	0.01	0.00	0.00
$H^N_{(8,256)}$	0.62	1.00	0.00	1.00	0.00	0.15	0.00	0.00	0.03	0.00
$H^N_{(16,256)}$	1.00	0.62	0.00	0.37	0.00	0.39	0.00	0.02	0.00	0.15
$H^N_{(32,256)}$	1.00	1.00	0.00	0.25	0.00	0.79	0.00	0.00	0.18	0.06
$H^N_{(64,256)}$	1.00	1.00	0.00	0.00	0.00	0.24	0.00	0.00	0.48	0.00
$H^N_{(2,512)}$	1.00	1.00	0.00	0.22	0.00	0.07	0.50	0.50	0.00	0.68
$H^N_{(4,512)}$	0.72	0.37	0.00	0.05	0.00	0.05	0.00	0.03	0.00	0.01
$H^N_{(8,512)}$	0.62	1.00	0.00	1.00	0.00	0.72	0.00	0.05	0.15	0.00
$H^N_{(16,512)}$	0.48	1.00	0.00	0.04	0.00	0.39	0.00	0.15	0.00	0.41
$H^N_{(32,512)}$	1.00	1.00	0.00	0.07	0.00	0.22	0.00	0.01	0.29	0.05
$H^N_{(64,512)}$	0.75	0.62	0.00	0.48	0.00	0.55	0.00	0.00	0.48	0.00
$H^N_{(2,1024)}$	0.75	0.22	0.23	0.62	0.00	0.07	0.00	1.00	0.00	0.00
$H^N_{(4,1024)}$	0.68	0.48	0.00	0.02	0.00	0.68	0.00	0.01	0.00	0.00
$H^N_{(8,1024)}$	1.00	0.48	0.00	1.00	0.00	1.00	0.00	0.01	0.02	0.16
$H^N_{(16,1024)}$	1.00	1.00	0.00	0.25	0.00	0.29	0.00	0.02	0.00	0.01
$H^N_{(32,1024)}$	0.62	1.00	0.00	0.07	0.00	0.25	0.00	0.00	0.00	0.00
$H^N_{(64,1024)}$	0.68	1.00	0.00	0.00	0.00	0.18	0.00	0.00	0.48	0.00

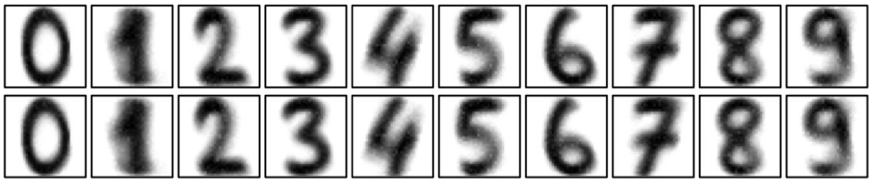

Fig. 11. Original mental images (top) and clone mental images (bottom).

and fixed by the number of tics t used to discretize the features values, and the number of dataset features F. On this two-dimensional image ($t \times F$), we draw gray-level pixels representing the feature value frequencies in the training set.

We experienced that performance degradation occurs only at first cloning from the original system: successive cloning processes from the first clone system do not imply further degradation of performance as long as the same t is adopted. The complete set of mental images generated by the DRASiW devoted to *optdigits*[11] is reported in Fig. 11.

5. Conclusions

In the context of Weightless Neural Networks, this paper presents a new methodology to transfer the knowledge acquired by a source neural system to a target one. The work is limited to DRASiW systems although it can be extended to other RAM-based neural models. The transfer method is uniquely based on processing an internal data structure representing the current status of the neural network on the acquired knowledge, that we call "mental image". The information stored into the mental image is used to graft (through memory transfer) the acquired knowledge of the source system into a target system in such a way to reproduce in the average good prediction capability within an acceptable loss of accuracy. Indeed, the statistical analysis proved that by averaging on the benchmarks datasets the measurements of accuracy of original and clone system with the same configurations, the cloning process does not produce significant loss in accuracy.

The advantage of this approach is that the source and the target DRASiW systems might have different architectures in terms of bit-addressing of RAMs and, thus, in terms of number of neurons. The limitation of the memory transfer here proposed is that the target systems (i.e. the *clone* systems) have to operate in either the same or in a strongly related domain. In this light, our mechanism is related to the classical *transfer learning* topic (as considered in the machine learning community) although it does not completely conform to it.

Possible applications of the proposed memory transfer mechanism are in the area of multi-robot distributed control systems: a DRASiW-based robot can transfer the acquired knowledge about the working environment when, for instance, it has to move to another area and be replaced by another robot. Although the substitute robot could find a slightly different environment, it can use the hired knowledge as a bootstrapping model of the current domain in such a way to be soon ready to act in the new environment.

[11] We report only *optdigits* mental images because their silhouettes are easily interpretable.

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087

Another possible application in mobile computing is the use of DRASiW-based smartphone apps in which the DRASiW system is implemented as an empty WNN but with the best architecture to fit the available smartphone resources. Each time the system has to be used in a new domain, the downloading (*transfer*) of the proper knowledge (*memory*) related to the application domain, will produce a brand new WNN system (a clone of the original one) working in this new domain with performance comparable to the original one and with an architecture compatible with that of the selected smartphone.

Apart from the potential applications, the contribution of this work is mainly theoretical, since we have proposed a methodology for WNN memory transfer based on the processing of the information stored in the neural network internal status. At the same time, the transfer mechanism has been experimentally evaluated with good and promising results by comparing the classification capabilities of original and clone DRASiW system on several datasets in machine learning domain. In addition, as a future work, we are trying to merge the memories of two DRASiW systems in order to have just one system with new functionalities. More in detail, we are addressing the problem of grafting the memory of a DRASiW system in part of the memory of another already trained system without catastrophic forgetting.

References

[1] G. Towell, J. Shavlik, Extracting refined rules from knowledge-based neural networks, Mach. Learn. 13 (1) (1993) 71–101.
[2] L. Pratt, J. Mostow, C. Kamm, A. Kamm, Direct transfer of learned information among neural networks, in: Proceedings of AAAI-91, 1991, pp. 584–589.
[3] L. Bochereau, P. Bourgine, Extraction of semantic features and logical rules from a multilayer neural network, in: International Joint Conference on Neural Networks, vol 2, 1990, pp. 579–582.
[4] S. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[5] S. Thrun, L. Pratt (Eds.), Learning to Learn, Kluwer Academic Publishers, Norwell, MA, USA, 1998.
[6] D. Silver, Selective transfer of neural network task knowledge (Ph.D. thesis), University of Ontario, 2000.
[7] C. Kandaswamy, L.M. Silva, L.A. Alexandre, R. Sousa, J.M. Santos, J.M. de Sa, Improving deep neural network performance by reusing features trained with transductive transference, in: Artificial Neural Networks and Machine Learning ICANN 2014, Lecture Notes in Computer Science, vol. 8681, Springer International Publishing, 2014, pp. 265–272.
[8] C. Kandaswamy, L.M. Silva, L.A. Alexandre, R. Sousa, J.M. Santos, J.M. de Sa, Improving transfer learning accuracy by reusing stacked denoising autoencoders, in: 2014 IEEE International Conference on Systems, Man and Cybernetics (SMC), 2014, pp. 1380–1387.
[9] N. Patricia, B. Caputo, Learning to learn, from transfer learning to domain adaptation: a unifying perspective, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1442–1449.
[10] R. Sousa, L.M. Silva, L.A. Alexandre, J.M. Santos, J.M. de Sa, Transfer learning: current status, trends and challenges, in: 20th Portuguese Conference on Pattern Recognition, RecPad, 2014, pp. 57–58.
[11] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, G. Zhang, Transfer learning using computational intelligence: a survey, Knowl.-Based Syst. 80 (2015) 14–23 (25th Anniversary of Knowledge-Based Systems).
[12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: a deep convolutional activation feature for generic visual recognition, CoRR, 2013, abs/1310.1531. URL ⟨http://arxiv.org/abs/1310.1531⟩.
[13] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, Overfeat: integrated recognition, localization and detection using convolutional networks, CoRR, 2013, abs/1312.6229.
[14] N. Srivastava, E. Mansimov, R. Salakhutdinov, Unsupervised learning of video representations using lstms, CoRR, 2015, URL ⟨http://arxiv.org/abs/1502.04681⟩.
[15] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? CoRR, 2014, URL ⟨http://arxiv.org/abs/1411.1792⟩.
[16] A. Giel, R. Diaz, Recurrent Neural Networks and Transfer Learning for Action Recognition, Technical Report, Final Project for CS231, 2015, Stanford.
[17] R. Andrews, J. Diederich, A.B. Tickle, Survey and critique of techniques for extracting rules from trained artificial neural networks, Knowl.-Based Syst. 8 (6) (1995) 373–389.
[18] L. Özbakir, A. Baykasoğlu, S. Kulluk, A soft computing-based approach for integrated training and rule extraction from artificial neural networks: difaconn-miner, Appl. Soft Comput. 10 (1) (2010) 304–317.
[19] M.H. Mohamed, Rules extraction from constructively trained neural networks based on genetic algorithms, Neurocomputing 74 (17) (2011) 3180–3192.
[20] J. Yu, C. Li, W. Hong, S. Li, D. Mei, A new approach of rules extraction for word sense disambiguation by features of attributes, Appl. Soft Comput. 27 (2015) 411–419.
[21] P.V.S. Coutinho, H.C.C. Carneiro, D.S. Carvalho, F.M.G. França, Extracting rules from DRASiW's "mental images", in: ESANN, 2014, pp. 529–534.
[22] B.P.A. Grieco, P.M.V. Lima, M. De Gregorio, F.M.G. França, Extracting fuzzy rules from "mental" images generated by modified WiSARD perceptrons, in: 17th European Symposium on Artificial Neural Networks (ESANN), 2009, pp. 25–30.
[23] M. De Gregorio, On the Reversibility of Multi-discriminator Systems, Technical Report 125/97, Istituto di Cibernetica, CNR, 1997.
[24] M. De Gregorio, M. Giordano, Memory transfer in DRASiW-like systems, in: 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), 2015, pp. 313–318.
[25] T.B. Ludermir, A. Carvalho, A.P. Braga, M.C.P. Souto, Weightless neural models: a review of current and past works, Neural Comput. Surv. 2 (1999) 41–61.
[26] I. Aleksander, M. De Gregorio, F.M.G. França, P.M.V. Lima, H. Morton, A brief introduction to weightless neural systems, in: ESANN, 2009, pp. 299–305.
[27] W. Bledsoe, I. Browning, Pattern recognition and reading by machine, in: Proceedings of the Eastern Joint Computer Conference, 1959, pp. 225–232.
[28] I. Aleksander, W. Thomas, P. Bowden, WISARD a radical step forward in image recognition, Sens. Rev. 4 (1984) 120–124.
[29] I. Aleksander, H. Morton, An Introduction to Neural Computing, Chapman & Hall, London, 1990.
[30] C.M. Soares, C.L.F. da Silva, M. De Gregorio, F.M.G. França, A software implementation of the WiSARD classifier (written in Portuguese language), in: SBRN '98, 1998, pp. 225–229.
[31] B.P.A. Grieco, P.M.V. Lima, M. De Gregorio, F.M.G. França, Producing pattern examples from "mental" images, Neurocomputing 73 (7–9) (2010) 1057–1064.
[32] E. Burattini, M. De Gregorio, G. Tamburrini, Generation and classification of recall images by neurosymbolic computation, in: ECCM98, 1998, pp. 127–134.
[33] C. Chang, C. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011) 27:1–27:27.
[34] H.C.C. Carneiro, F.M.G. França, P.M.V. Lima, Multilingual part-of-speech tagging with weightless neural networks, Neural Netw. 66 (2015) 11–21.
[35] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (6) (1945) 80–83.
[36] Q. McNemar, Note on the sampling error of the difference between correlated proportions or percentages, Psychometrika 12 (2) (1947) 153–157.
[37] A. Edwards, Note on the "correction for continuity" in testing the significance of the difference between correlated proportions, Psychometrika 13 (3) (1948) 185–187.

Massimo De Gregorio was born in Torre del Greco (Naples – Italy) in 1962. Upon completion of his graduate education (Laurea in Fisica – Cibernetica) at the University of Naples, in the 1994 he began working as researcher at the Istituto di Cibernetica of Consiglio Nazionale delle Ricerche (CNR) in the Research Group "Theory and Techniques of Knowledge Representation". Between 1999 and 2003, he held visiting teaching positions at University of Naples "Federico I" (2000–2003) and Second University of Naples (1999–2000). His research work has been mainly concerned with Artificial Intelligence, Artificial Neural Networks (in particular Weightless Neural Systems), Neuro-Symbolic Hybrid Systems, with Extensive applications in Expert Systems, Pattern Recognition, Active Video Surveillance and more recently in Behavior Based Robotics. He is a co-author of different invited papers of several international conferences on Artificial Intelligence and Artificial Neural Networks.

Maurizio Giordano was born in Naples on 6th of August 1966. In 1992 he graduated (cum laude) in Physics with specialization in Cybernetics at the University of Napoli "Federico II". From 1991 to 2000 he has been working at the Cybernetics Institute of CNR (CNR-ICIB) with grant support and temporary research positions. From 2001 he is a CNR research scientist with a permanent position. From 2000 to 2013 he has taught Operating Systems and Programming Languages at the Computer Science course of the Faculty of Science (University of Naples "Federico II"). His research activities and expertises are in the fields of High Performance and Grid Computing, Service-Oriented Architectures and Artificial Neural Networks. From 1996 he has participated to several European Projects: Nanos and VHF Esprit Projects FP4, POP LTR Project FP5, S-Cube (NoE) and MiDAS Projects FP7. From March 2014 he works at the High-Performance Computing and Networking Institute of CNR (ICAR-CNR). He is a coauthor of fifty papers published in international journals and conferences. He has been member of conference committees and evaluation boards for research grant and acquisition of HPC infrastructures.

Please cite this article as: M. De Gregorio, M. Giordano, Cloning DRASiW systems via memory transfer, Neurocomputing (2016), http://dx.doi.org/10.1016/j.neucom.2016.01.087